Automated nesting of sheet metal parts

Scott, Andrew James

Award date:
1996

Awarding institution:
University of Bath

Automated Nesting of Sheet Metal Parts

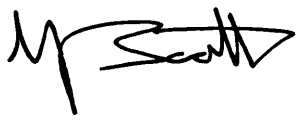
submitted by Andrew James Scott

for the degree of PhD

of the University of Bath

1996

COPYRIGHT



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Summary

Nesting is the positioning of two dimensional parts onto a sheet of material with the aim of minimising the waste in the layout. Nesting has traditionally been carried out manually by a planning engineer, however increasing effort is being applied to its automation.

Previous research has identified a number of strategies for the automation of the nesting process. One strategy is to enclose each part, or a cluster of parts, within a rectangular envelope. These envelopes can then be placed on a sheet without the detailed geometry of each enclosed part being considered. A new nesting system has been designed around this strategy, the core of which is the efficient placement of the rectangular envelopes. Two new algorithms to place rectangular envelopes, the Area Fit algorithm and the Strip Fit algorithm, have been developed. A third new algorithm, the Strip Squeeze algorithm, can be used to improve any layout which consists of strips of parts.

A new nesting system that incorporates the three new algorithms has been developed. The system, which is limited to the placement of rectangular envelopes or rectangular parts, also contains two established algorithms, which are included as performance benchmarks. All the algorithms have been tested across a broad range of nesting problems and the new algorithms have generally out performed the two established benchmark algorithms (Next Fit and First Fit). In addition, an investigation into the effects of part list size, part aspect ratio, sheet aspect ratio and sheet size on the performance of each algorithm has been carried out. All of these features have been shown to have a significant effect on the amount of waste incurred in layouts of parts.

Acknowledgements

I would like to thank my supervisor, Dr. Tony Mileham, for his encouragement and guidance throughout the course of my research and the writing of this thesis.

I am grateful to my colleagues within the School of Mechanical Engineering, in particular those with whom I have shared an office, for their help and support.

I would also like to thank my family and friends for their support and encouragement throughout my research.

This work has been funded by the Engineering and Physical Sciences Research Council (EPSRC).

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Scope and Aims of the Work

Sheet metal manufacture dates back to the ancient Egyptian civilisation, which developed complex hand forming techniques and the metal spinning process [Smith (1992)]. However, all sheet metal processes at this time were limited by the power which could be supplied by the operator or an assistant. The advent of power tooling during the industrial revolution removed this constraint and allowed the development of the vast range of sheet metal processes which exist today. Sheet metal products are encountered in all aspects of modern life. Domestic appliances, kitchen utensils and televisions all contain sheet metal parts, and almost all vehicle bodies are made from sheet metal. Products can vary in size from a huge wing section for a commercial aircraft, to the tiny delicate components for clocks and watches. Volumes of production can vary from a one-off part for a satellite, to the millions of food cans produced each day. Thus the range and variety of sheet metal products is immense.

Owing to the variety of processes, the planning of a sheet metal part's manufacture can be very complicated. Generally, the part will be cut from the stock material before any forming or bending processes are applied. However, formed features can be created before the part is cut from the sheet and this may be preferred when it improves the local sheet stability during the forming process. Also, high volume parts may have cutting, bending and forming operations carried out simultaneously by dedicated tooling.

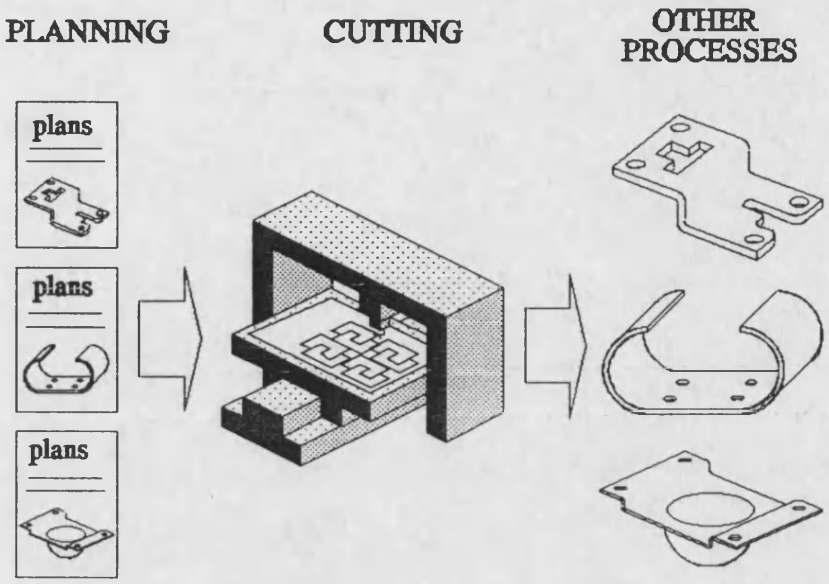


Figure 1.1.
The convergence of parts through the cutting process.

The process planning for most parts consists of applying a number of discrete processes to the part in a defined sequence with no consideration given to the process plans of other parts. One common exception to this is in low volume manufacturing environments where a capacity problem may exist on a machine shared by two parts resulting in one part being manufactured on a less favourable machine. In low volume sheet metal manufacture some interaction between different parts is common as they will often be cut from one stock sheet and thus will affect each others production directly, as shown in Figure 1.1. This creates an interesting and unusual process planning situation.

The scope of the work in this thesis is strictly limited to the problem of placing parts to be cut from sheets of metal in an efficient way and then using non-dedicated cutting methods such as guillotine shears, rectangular shears or flame cutting to separate them. The planning and layout problems faced when using tooling dedicated to a single part (eg. a blank) or cutting and creating formed features or bends simultaneously will not be considered. When placing parts on a sheet the objective is to produce a 'tight' or 'dense' layout to minimise the material waste and the tool travel. Figure 1.2 shows an artificial 'perfect' layout of 25 parts which contains no

unavoidable waste and Figure 1.3 shows a layout of the same parts achieved using a poor algorithm. Creating such layouts is generally known as 'The Two Dimensional Layout Problem' or 'The Part Nesting Problem'. There is considerable similarity between planning the cutting of sheet metal and that of other sheet materials such as paper, cloth, leather and glass. However, this work is focused on sheet metal so some of the layout requirements particular to other materials may not have been considered.

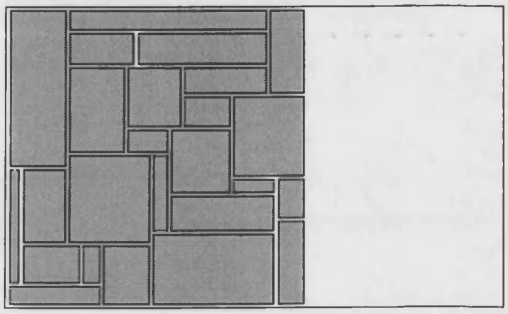


Figure 1.2
A Perfect Layout.

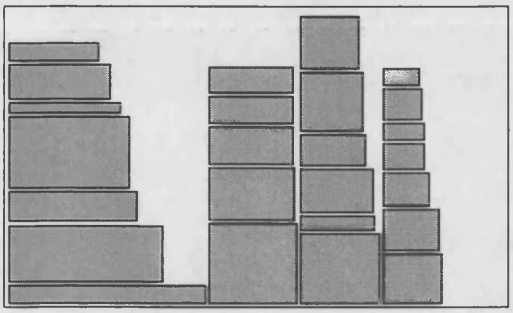


Figure 1.3.
A Poor Layout.

Figure 1.4 shows the general stages involved in the production of a sheet metal part from design to manufacture. The nesting problem is a major element in the detailed process planning for the cutting of sheet metal parts. From the figure it can be seen that detailed planning is the last stage of process planning.

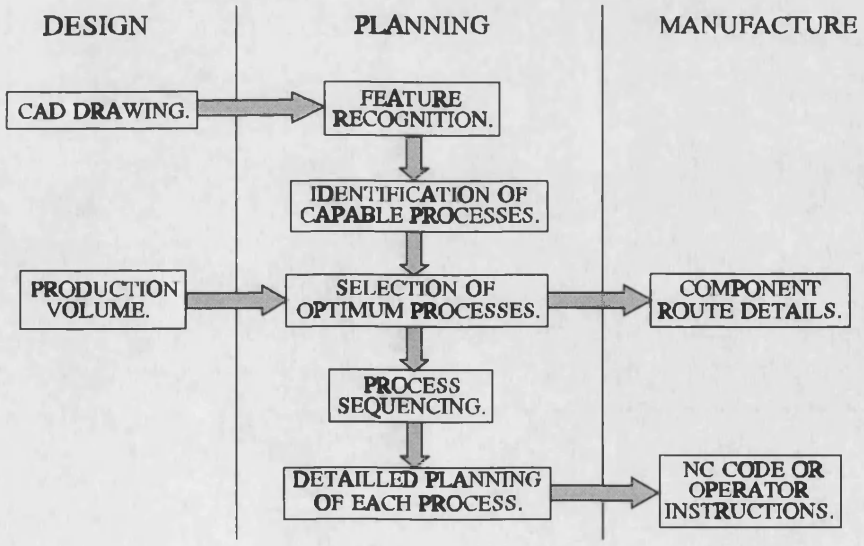


Figure 1.4.
Stages of Process Planning.

To summarise, the aims of this research are:-

- To design a new automated nesting system which creates layouts with less waste than layouts created by the existing nesting systems.
- To target the new nesting system at the 'low volume - high variety' production environment.
- To develop the core 'part placement' element into a working system.
- To test this new 'part placement' element of the system against comparable existing systems.
- To evaluate the effect of various nesting problem parameters, such as sheet size, on the performance of this part placement element.

The restrictions on this research are:-

- To not consider the problem of blanking a large number of identical parts from a strip of material.
- To not tackle the import of part data from a CAD source.
- To not tackle the subsequent export of data to a cutting path planning system.

During this research, work has been carried out in the field of process selection for sheet metal manufacture which will not be covered in this thesis. This covers the second and third stage of process planning shown in Figure 1.4. Scott and Mileham (1993b) contains a comprehensive review of all sheet metal manufacturing processes and has classified them into the categories of cutting, bending and forming processes. The process selection system developed [Scott and Mileham (1992) & (1993b)] relies on the identification of 'process critical' features which force one particular process or process type to be used. This system identifies all the manufacturing processes capable of producing the part's features, from which the optimum processes can be selected on the basis of economics.

1.2 Cutting Sheet Metal

There are three general process groups involved in the manufacture of sheet metal components; cutting, bending and forming. Within each of these groups there is a considerable range of individual processes, some of which overlap in the types of features they can produce. As this work is only concerned with the planning of sheet metal cutting, the processes to bend and form sheet metal will not be covered. The individual bending and forming processes have been described in detail in earlier work [Scott and Mileham (1993b)].

The traditional method of cutting sheet metal is **Punching**; this is a general term for the mechanical shearing through a sheet of material in a single blow. Within this general process there are six types of punching, identified by Lascoe (1988), which are:-

(1) **Perforating** is the punching of a hole of any profile within the parent material.

(2) **Blanking** is the punching of a required external profile shape from a sheet.

(3) **Notching** is the punching of a shape in the perimeter of the parent material.

(4) **Nibbling** is the repeated punching of an area to produce a 'Perforation' or a 'Notch' of greater area than the tool used. Nibbling is used when a tool of sufficient size or the required profile is not available or the press is not powerful enough to cut the required profile in a single strike.

(5) **Lancing** is the punching of only a section of a shape's perimeter. Thus leaving the shape attached to the parent sheet but protruding from its face. This feature is generally used as an interface location, a stop or a spring.

(6) **Piercing** a hole is not strictly a punching operation, as the material is deformed rather than removed. The deformed material strengthens the hole, which is ideal if it is to be tapped. Existing holes can also be 'Extruded' to a greater size by forcing a mandrel through them; this produces a more uniform deformation. Tools are available which punch a small hole and then extrude it to the required size in one strike.

In general a punch tool consists of a punch, a die and a stripper, as shown in Figure 1.5. The punch applies force and the sheet is sheared between it and the die. The clearance between the punch and die must be accurately calculated for the sheet gauge and the material type, otherwise excessive cutting forces will be required and a poor edge finish produced. The stripper holds the sheet between it and the die, preventing movement and local deformation of the sheet around the feature being cut.

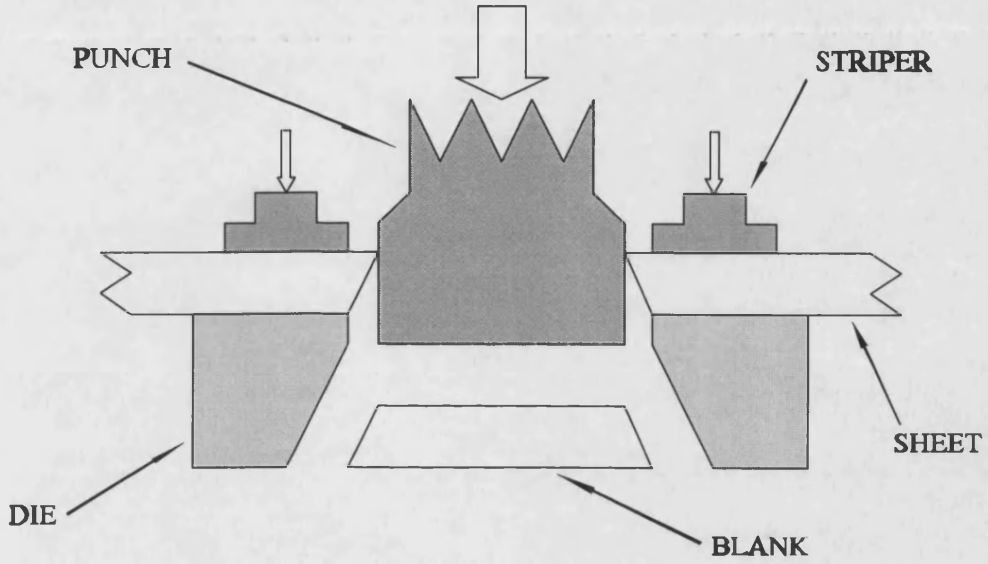


Figure 1.5.
Sheet Metal Punching.

The shear failure edge of a punched hole is not perpendicular to the sheet face, but lies at an acute angle to it. Thus the size of the blank does not correspond to the size of the hole. The punch dictates the size of the hole and the die dictates the size of the blank. Punching features with sharp corners is problematic as the excessive force concentrations lead to premature tool wear and/or failure. The cutting force required by each edge segment of a feature must be calculated and combined to give the force centre of the feature, which must lie on the axis of the punch. If this is not achieved the punch will deflect when cutting, again leading to poor finish and premature failure of the tool. Thus punch tools are complicated and relatively expensive, suiting them to high volume manufacture of a single component type.

Components which are large and complicated are rarely 'blanked', as a single strike cut would require excessive press power and an exceptionally complicated and expensive tool. Thus large components are often cut from sheets in stages. The traditional method is to use **Guillotine Shears**. These shears span the entire sheet and can only cut in straight lines, this is known as the 'Guillotine Cut Constraint'. This demands that components to be cut from a sheet by this method are placed in rows and columns which span the length or width of the sheet. However, these rows only need to span the sheet as they are about to be cut. Thus a row may terminate on a perpendicular row of components, providing the latter row precedes the former row in the order of cutting, see Figure 1.6 and Figure 1.7. As one side of the sheet is sheared in the unconventional direction the angle of the cut face to the sheet face will be obtuse rather than acute. However, this is unlikely to cause problems unless a precise edge is required on a sheet with a special surface finish on one side. In addition, unless a straight edged shape with no concave elements is required, the part will require further processing.

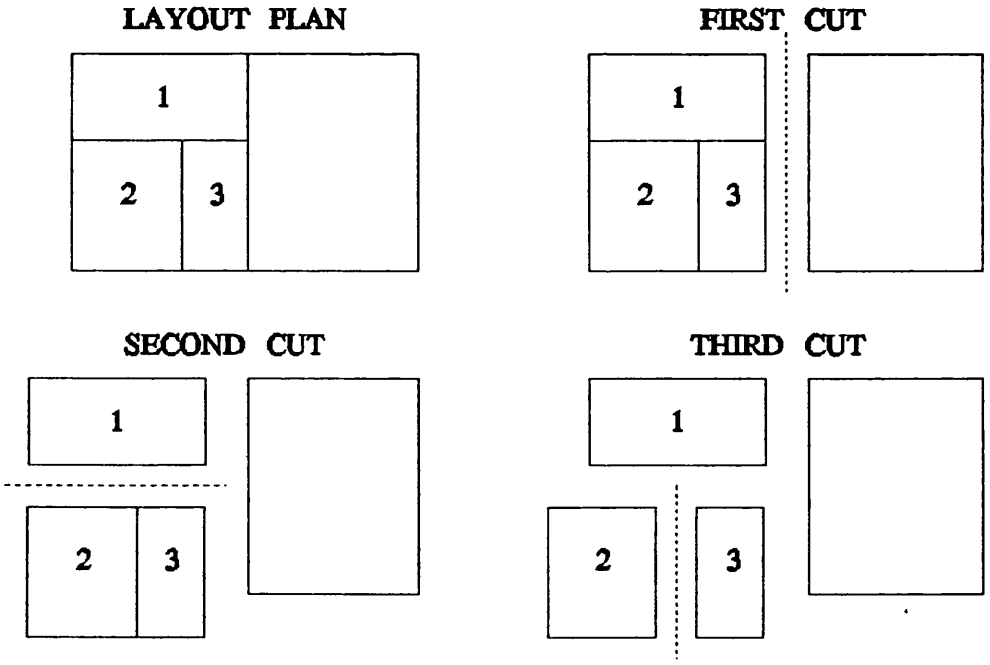


Figure 1.6.
A layout which conforms to the 'Guillotine Cut Constraint'. The cuts must be made in this sequence.

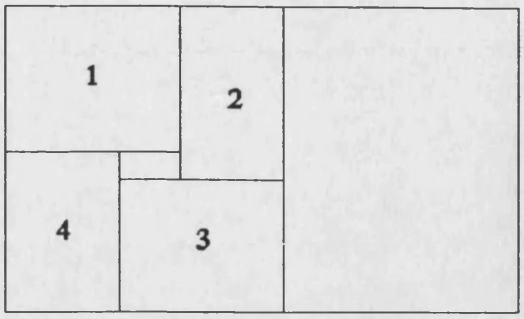


Figure 1.7.
A layout which does not conform to the 'Guillotine Cut Constraint'.

Rectangular Shears, which punch out a long thin strip of metal, can be used in a similar way to nibbling to cut out straight edge component perimeters, see Figure 1.8. This method avoids the guillotine cut constraint and is therefore more flexible, but wastes thin strips of sheet. Unlike the Guillotine method both cut faces are at the conventional acute angles.

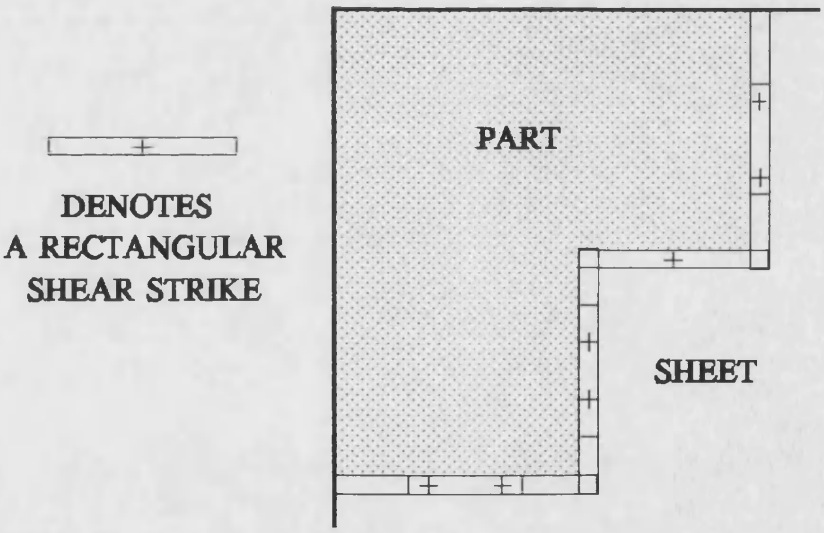


Figure 1.8.
Cutting with Rectangular Shears.

A less common process is **Slitting with Rotary Knives (Scroll Shears)**, described by Kalpakjian (1989), which is used in the can industry. This consists of a pair of interlocking circular blades which are rotated as the sheet is passed between them. This method can create arcs of a reasonably small radius, as well as straight cuts, however the exact limits of the arc cutting capability will depend on the diameter of

the blades and the thickness of the sheet. This process, like conventional shearing, has no associated waste generation.

Sheet metal can also be cut by a variety of 'flame' cutting processes that use mechanically similar control systems for sheet manipulation. **Gas Cutting** is the slowest process; the high heat input, and hence distortion, makes it inappropriate for thin plate or sheet. Thus **Laser Cutting** and **Plasma Cutting** are favoured for such materials. Plasma Cutting has a particularly high cutting temperature giving limited heat distortion and allowing high cutting speeds [Salisbury (1991)].

Flame cutting machines consist of a large bed, onto which the sheet is positioned and the cutting head mounted on a bridge which can traverse the length of the bed. The system can be NC controlled, thus co-ordinate programs can be entered to allow complex contours to be cut from the sheet. Due to the rapid movement required of the control system, laser and particularly plasma cutters require powerful servo motors and a light, but rigid machine structure. Thus their capital costs are high.

The greatest advantage of flame cut systems is they use no dedicated tooling and are therefore far more cost effective than punch and press systems for small batch production. However, punch and press systems have by far the shorter cycle time, as many features can be cut with a single press blow. Thus they are the best systems for high volume production where the capital cost of dedicated tooling can be recovered. Flame cut systems also have advantages over punch and press systems in their ability to cut complicated profiles quickly with no requirement for dedicated tooling.

Intricate cut patterns and small holes can cause problems for laser and plasma cutting methods for two reasons. If the flame has to dwell in one area for some time to produce a complicated feature sufficient heat may be put into the sheet to cause distortion. For small features it is difficult for the control system to achieve the accuracy which a punch would produce. One solution to this is the combination of a laser cutter and an NC punch press into the same machine, allowing the advantages of both methods to be combined.

Flame cutting and rectangular shears require a 'Bridge Gap' to be placed between components when they are nested onto a sheet, see Figure 1.9. This is due to the material which is removed or destroyed through the method of cutting. For some products the sides of the sheet may have an unsuitable finish, which requires a peripheral bridge gap to be placed between the edge of the sheet and the parts as in Figure 1.10. Small fingers of material are often left with these methods to hold a component in the skeleton of its sheet, this allows the sheet to be removed from the bed of the cutting machine as a whole. The fingers of material are made sufficiently thin to allow the components to be snapped out of the skeleton sheet. This method is particularly popular when a 'kit' of parts for a complete product are located on one sheet. This allows simple part management and parts can be removed from the sheet when required for subsequent operations or assembly.

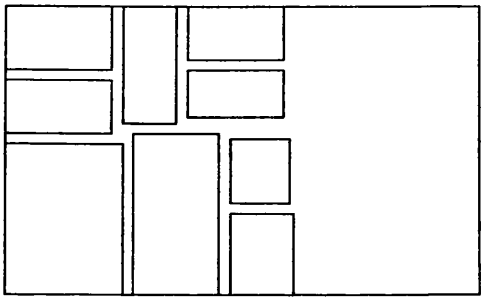


Figure 1.9.
Bridge Gap between parts.

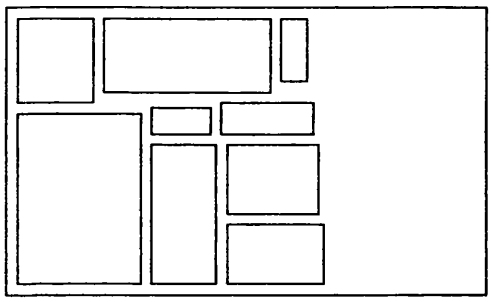


Figure 1.10.
Sheet perimeter Bridge Gap.

Bending and forming operations cannot be carried out on a part at the same time as cutting operations without employing special dedicated tooling. As this type of manufacture is beyond the scope of this work bending and forming processes are not examined. Subsequent bending and forming operations may require additional material to be attributed to a part to allow for clamping or to provide local stability. However, this can be accounted for before the part shape is considered for cutting and therefore has no direct effect on layout planning.

1.3 Creating Sheet Layouts (Nesting)

Nesting is the positioning of a number of shapes or entities within a defined space with the aim of minimising the area which they occupy. In the sheet metal environment nesting is constrained to locating two dimensional shapes on standard sized sheets of material. Further constraints may also be imposed, for instance certain components may have to be grouped together for ease of production control. Some components may also have a pre-defined orientation, for instance, bends which are to be subsequently made in a component may not be permitted to lie 'in line' with the sheet's grain, for strength reasons. The efficiency of the nesting system determines the level of waste generated during manufacture. The traditional cutting method of Guillotine Shearing confines nesting patterns to those formed by cuts spanning the width of the sheet. This constraint limits nesting efficiency, but simplifies the nesting problem. The development of rectangular shears and laser cutting has removed this constraint from many production environments.

Traditionally nesting has been carried out by a Planning Engineer using informally gathered expertise to position scale models of components into an area corresponding to that of the stock sheet, in such a way that the waste is minimised. In developing any new system it is useful to examine the system which it will replace. Thus it is of value to examine the rules and methodologies which human planners use. The abilities of a human planner must also be evaluated against those of an automated system to recognise their relative benefits and efficiencies.

Humans have a highly developed spatial awareness and strong powers of geometric reasoning which cannot yet be mimicked by computers. This has been developed as a response to the three dimensional real environment and the day to day tasks which are carried out within it. Nesting is predominantly concerned with making close shape matches between component perimeters. A human planner can glance at a component and imagine it rotated and mirrored in relation to another component to find the orientation which gives the best match. Humans also have a good ability to recognise visual patterns which match, for instance when completing a jigsaw puzzle.

Although computers are continually expanding in power, it is considered that it will be some time before they can match the performance of human planners in the environment of geometric or spatial problems. Thus, rules are required to simplify the complex geometric and spatial elements of nesting to manageable proportions. The manipulation of two dimensional shapes in accordance with rigidly defined simple rules is within the capabilities of todays moderately priced computers, allowing systems which are developed to be of immediate practical value.

One clear disadvantage in manual layout design is the inability of a human planner to consider more than a small number of parts, say 5-10, at one time. This generally results in the components being nested in an arbitrary sequence, unless a vast amount of time is available to allow a trial and error approach. This is an area where a computer has the potential to out perform the human planner by repeatedly searching the entire parts list, particularly if the list is large. Human planners often simplify a large layout problem by creating clusters of parts which can then be manipulated as single entities. This is often through recognising that a group of parts will form a neat strip or an efficient repeated pattern. These simplification methods should be considered in an automated system as they can reduce the problem whether it is being tackled by a human planner or a computer.

Adding the cut-allowance to the perimeter of a component is a time consuming task manually, but can be carried out quickly by an automated system. Also a human planner cannot accurately quantify how efficient a nesting layout is without considerable time spent calculating or estimating the areas of waste on a layout. This is another area where an automated system would excel.

1.4 Intractability and NP-Completeness

It is useful to consider the mathematical complexity of a problem when looking to develop an algorithm to solve it. This is generally carried out using the theory of NP-Completeness [Garey and Johnson (1979)]. Any algorithm with a time complexity function which cannot be defined as or bounded by a polynomial expression is considered to be NP-Complete (ie. it is a Non-deterministic polynomial time algorithm). It is not possible to mathematically prove a problem is NP-Complete. Problems are shown to be NP-Complete indirectly, by proving that they are of the same order of complexity as a known NP-Complete problem.

Some NP-Complete exponential time algorithms have been shown to run quickly in practice eg. the branch-and-bound algorithms for the knapsack problem [Albano and Orsini (1979a)]. However, generally they are impractical and an alternative route is required to provide a solution. Common methods are to further constrain the problem by, for example, accepting the first solution to exceed a threshold acceptance level or developing heuristic rules to give a 'good' solution. All of these methods compromise the optimality of the solution.

Turing (1936) identified a number of 'undecidable problems', for which it was proven that no algorithm could be designed which would generate a solution known to be optimal. This is the case for the unconstrained layout problem due to the infinite range of positions and orientations which each part could adopt. If the parts are constrained to a fixed orientation and defined part placement rules are adopted, thus the only freedom remaining is the sequence of part placement. However, if n parts are to be placed there are n! possible solutions (shown graphically in Figure 1.11). This is of a similar order to the Travelling Salesman Problem which is known to be NP-Complete. If the parts can be placed in their primary orientations (0° or 90°) the number of possible solutions is increased by a factor of 2^n, the effect of which is also shown in Figure 2.11.

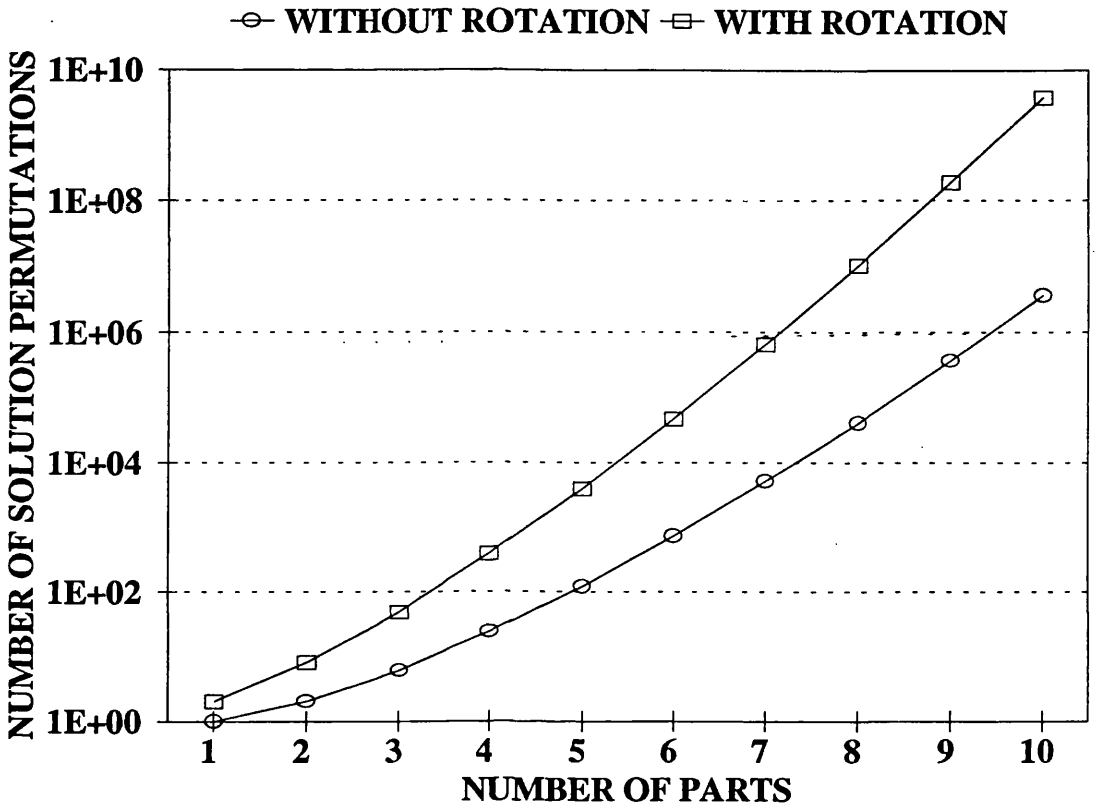


Figure 1.11.
The relationship between the number of parts and the number of solutions to an NP-Complete approach to the layout problem.

1.5 Process Planning of Sheet Metal Cutting

Wang and Wysk (1987) give a simple definition of process planning as *"the act of preparing detailed work instructions to produce a part"*. The traditional method of process planning is for a human planner to select a suitable sequence of machines and write detailed instructions for the operator, or if appropriate, an NC program for the machine.

Many researchers have formalized the steps of process planning, either in general terms or specifically for a range of products and a manufacturing environment. Smith et al (1992) define a three-step procedure given below to be the essence of process planning.

(1) Determine systematically the configuring operations required to produce the part.
(2) Identify the ancillary operations required to produce the part.
(3) Place operations in their optimal and logical sequence for the fabrication process.

Alting and Zhang (1989) have defined in greater detail 10 stages of process planning. Although this is a general list of factors, all are relevant to the specific planning of sheet metal cutting.

(1) Interpretation of the product design data.
(2) Selection of machining processes.
(3) Selection of machine tools.
(4) Determination of fixtures and datum surfaces.
(5) Sequencing the operations.
(6) Selection of inspection devices.
(7) Determination of production tolerances.
(8) Determination of proper cutting conditions.
(9) Calculation of the overall times.
(10) Generating the process sheets including NC data.

The planning of sheet metal cutting can be divided into four clear stages.

(1) Gathering the design information for all the parts to be cut from a single gauge and type of sheet metal.
(2) Importing the part's information into the nesting system. The data is most likely to be in the form of a number of part drawings from a CAD system. This is covered in greater detail in Section 1.6.
(3) Creation of an efficient layout ie. the 'part nesting' or 'two dimensional layout' problem.
(4) The layout produced must be converted into planning sheets of cut sequences and tooling or a program for an NC cutting machine. This is covered in greater detail in Section 1.7.

According to Wang and Wysk (1987) most planners have gained their process knowledge through years of experience as machine operators. Thus planners are rarely trained and are generally appointed on the basis of their abilities as operators. Planning by untrained planners has a number of associated problems, for example process technology changes are often not accounted for if the planner left the production environment before the changes were introduced. Even if training is given in a new process the planners will feel more comfortable with those processes with which they have 'hands on' experience. Planners may also favour machines with which they have the most experience even if the application is inappropriate. This results in a lack of standardisation and repeatability in the process planning function. Conflict in approaches can arise when two planners are required to collaborate on the planning of a complicated component requiring their combined expertise. A US Air Force survey (Wang and Wysk 1987) estimated that US industry requires 200 000 to 300 000 planners, while only 150 000 to 200 000 are available. For these reasons there has been considerable investment in Computer Aided Process Planning (CAPP) systems.

A CAPP system is one which automates or aids any stage of process planning and can take any of the following general formats. A **Variant Planning System** retrieves the

process plan of a similar component from a database. This must then be adapted by an experienced planner, which may inhibit fresh thinking on process planning problems and also prevent new technology or methods being applied to established products. **A Semi-Generative Planning System** retrieves the process plan of the nearest existing component and adapts it to suit the new component. This is quicker than generative planning but prevents the entire component from being considered as a single entity, which may compromise the plan's efficiency. A **Generative Planning System** applies data and rules to make planning decisions without reference to any other component's plans. This method is only viable if clear rules can be formed which produce effective process plans.

A layout consisting of parts similar to those to be nested could be identified and retrieved from a database of layouts. However, the layout could only be easily adapted if all the parts to be removed were larger than their substitutes. This would considerably compromise the layout efficiency. Another method would be to keep a database of combinations of parts which form tight clusters. Appropriate clusters could then be retrieved to aid future layout design. Neither of these approaches seem very promising. As the bounds and objectives of the layout problem can be clearly defined it is a suitable task for a generative planning system.

Many prototype generative planning systems that cover small knowledge domains have been developed in the form of Expert Systems. Unlike traditional software programs, Expert Systems hold factual information and decision rules in separate areas. This allows discrete areas of the system to be up-dated or changed without interfering with the global structure and function of the system. Expert systems are structured into three sections: a Declarative Knowledge Base of factual information and production rules, an Inference Engine of rules and procedures to select and use the relevant declarative knowledge and the Metaknowledge section which defines the overall strategy by which the inference engine operates.

Gupta and Ghosh (1988) have carried out a survey of expert systems in manufacturing and process planning. This identifies applications as diverse as optimising rivet

configurations for joining steel plates, designing new aluminium alloys and optimising a glass annealing process. Gupta and Ghosh 1988 define three criteria which identify situations to which expert systems are ideally suited. These are that the problems in the domain cannot be well defined analytically, the number of alternative solutions is large and the domain knowledge is vast, requiring relevant knowledge to be used selectively.

The layout problem can be reasonably well defined analytically and the domain knowledge is not so large as to require selective use. A normal heuristic or algorithmic approach would seem to be more suitable for the particular problems faced in creating layouts of parts. An expert system approach may be more useful in examining the parts list and selecting particular algorithms, or fine tuning their operation, for the particular circumstances of each parts list.

Automation of the process planning function gives consistent repeatable planning decisions and aids process and tooling standardisation. Currently many sheet metal parts are designed using Computer Aided Design (CAD) systems and it is foreseeable that in time all components will be designed using CAD systems. CAPP systems exist which will convert a layout of parts on a sheet into a sequence of operations for an NC cutting device. Automation of sheet layout design will link these two areas to fully automate the planning of sheet metal cutting.

1.6 Data Input For Nesting

If the drawings of the parts to be nested are on paper rather than held as files in a CAD system the nesting methods which can be used are severely limited. Invariably this situation dictates manual layout creation by an experienced planner as described in Section 1.3. The only exception to this is if the drawings can be scanned into a CAD system or directly into a nesting system.

If the drawings are available in a CAD system it is desirable to automatically interpret them to obtain the information required for nesting. However, no single dominant CAD data format has emerged from the large number currently in use and none of the existing formats are clearly the most suitable for automated feature recognition and interpretation. Of the current formats IGES (Initial Graphics Exchange Specification) and DXF (Drawing Interchange File) are the most commonly used. However, neither can automatically link the relevant tolerances to the co-ordinate information. Linardakis and Mileham (1993) are developing methods of interpreting 3-D DXF files and linking in the tolerance information. This has been carried out for $2^1/_2$-D rotational parts [Zhang and Mileham (1990)]. Thus a dedicated interpreter may be required to transfer the data held in a CAD systems files into the planning systems. This interpreter may not be general enough to accommodate data entry from a different CAD system. All CAPP systems which need to manipulate data directly from CAD systems at present have this problem.

Currently the STEP (**St**andard for the **E**xchange of **P**roduct Model Data) standard CAD and Solid Model format is being developed [Qiao et al (1993)]. This format aims to allow simplified CAD-CAPP communication and the direct transfer of data from one drawing system to another by combining the tolerance and co-ordinate data. This standard is almost complete, but is not yet commercially available.

The nesting problem in the sheet metal environment is constrained to two dimensions, which is relatively simple for automated interpretation and manipulation. However, many parts to be cut will have bends and formed features which require three

dimensional representation. As the nesting system is one element of a desirable totally automated planning system, it may have to interpret three dimensional representations of two dimensional shapes to be cut. The only alternative would be for two drawings to be used, one of the finished part and one of the unfolded flat cutting pattern. Components are traditionally represented by lines and arcs in a 2-D format, with multiple view conventions to allow 3-D representation. Humans have an ability to conceptualise these representations into a 3-D component, and comprehend manufacturing features and design intent, beyond any current automated system's capabilities.

To interpret a drawing of a three dimensional part into it's two dimensional form so that it can be cut from the sheet, all bends and formed features must be removed. There are currently a number of CAPP systems which do this, although no systems aim to plan all sheet metal products. The most comprehensive systems currently documented plan components which can be completely manufactured using a limited range of cutting and bending processes, and no forming processes. Kujanpää and Penttilä (1992) describe a workstation based system which aims to totally integrate the Design, Planning and Manufacturing for all sheet metal components within the above constraints. The system has been built around commercial packages. Component data is entered through a Vertex CAD package and fed to a Radan CAM package via an IGES standard file. The system can interface with a company's Production Planning System (PPS) system. Components of common raw material type are automatically grouped for nesting. Limited information is given on the detailed operation of the system for reasons of commercial confidentiality.

Nnaji et al (1991) have developed a similar system with an unspecified CAD source supplying component data via IGES. Component unfolding is achieved by identifying the loop of B-Rep edges which contains at least one edge in the same plane as all of the other edge loops. This is the component perimeter. The folds are then flattened around their neutral axes using a database containing the necessary bend off-set data. The feature database has a hierarchical structure and contains the necessary process data to plan all cut features and all but a few bend features.

Smith et al (1992) describe the FCAPP/SM system which selects all the tooling, machines and operations necessary for, or capable of, the production of each feature. This information is entered into a relational database and compared to the process hierarchy shown in Figure 1.12 for sequencing. In all cases slots and holes on any section of a component with a bend are produced after the bend. Algorithms are employed to optimise machine tool movements and optimise the components production route. It is considered that rigid adherence to the first three stages of this hierarchy would result in excessive tool travel if either a laser cutting machine were used or a number of parts were to be simultaneously produced.

(1) Generate periphery.
(2) Generate pre-bend holes.
(3) Generate Pre-bend slots.
(4) Generate bends.
(5) Generate post-bend holes.
(6) Generate post-bend slots.

Figure 1.12.
Process hierarchy defined by Smith et al (1992).

In a survey which considered 156 CAPP systems, Alting and Zhang (1989) described 14 systems in detail. Of these only one German system, AUTAP, attempts to automate the planning of sheet metal components, and this was limited to a single company's products for the telecommunications industry. This highlights the lack of CAPP systems applied to the sheet metal manufacturing environment. This may be due to the range of sheet metal processes and the complexity of forming operations in relation to metal cutting processes. This survey concluded that the achievements of CAPP systems are poor considering the quantity of research which has been carried out in this field. This limited success is attributed to two sources: the problem of manipulating and interpreting dimensional tolerances in CAPP systems and the lack of a standardised format for CAD information which can be easily interpreted into manufacturing features.

1.7 Data Output From Nesting

The form of output from a nesting system is influenced by the subsequent cutting process to be used. The Guillotine method will require a simple layout of straight line cuts and their distances from the datum edges which will then be interpreted manually by the guillotine operator. This stage of cutting simply aims to break up the sheet into segments of the correct size for each part. These will then pass individually through other cutting processes to create each part's detailed features. The layout problem for the guillotine cutting method requires only a simple output of straight lines and locations which can easily be produced by any manual or automated system.

Another type of cutting is a crude flame cutting device (ie. oxyacetylene) which cannot produce finished features, but can approximate a part's form more accurately than a guillotine. Such a machine may be NC controlled requiring a cutting path program to be written or it may be manually controlled. Generally, manual control devices consist of a control pin which is traced around a scale drawing of the part. The cutting head mimics the control pins movement via direct mechanical linkage or servo motors. Occasionally, the sheet itself may be painted or etched with the cutting pattern which is then followed manually. This type of cutting requires greater part detail to be entered into the nesting system and also requires a more complicated output. However, the output required only consists of simple shapes with quite loose tolerances due to the accuracy of the process.

The most sophisticated NC cutting devices, which may be a laser or plasma cutter, an NC punch press or a combination of both, require much greater detail in the sheet layout. Many of these machines will carry out all of the cutting, requiring accurate information on the parts' features and their tolerances. To improve laser and plasma cutting efficiency the overall travel and changes of direction must be minimised. However, a dense cluster of features may have to be visited twice to prevent excessive heat input to that area of the sheet. NC punch press efficiency is improved by reducing tool changes and cutting head travel. Thus features common to more than one part must be identified and cut as a group to minimise tool changes. This type

of cutting requires all part features and tolerances to be obtained from the individual CAD files and combined into a single plan of the entire sheet layout. A number of systems exist to convert this data into instructions for NC cutting tool controllers.

Svestka et al (1981) aimed to improve the productivity of NC punch presses by tackling the 'travelling salesman' type problem of optimising the sequence of punch hits. The 'closest unvisited city' heuristic proved unsatisfactory, so an 'assignment' approach was attempted. This operates by creating a number of sub-tours between closely spaced hits, and then combining the sub-tours to give the complete tool route. This heuristic out performed the former in 90% of the trials. Raggenbass and Reissner (1989) describe a process planning system for flat sheet parts which contains a more modern system to tackle this problem. Tasks include the selection of machines, punches and laser cutting operations with the aim of optimising the total time and the cost of manufacture.

Hancock (1989) describes a planning system developed for the Otis Elevator Company. The products in question are of standard forms, but an almost infinite variety of part dimensions are possible. The system converts AutoCAD drawings into a list of manufacturing features. The system minimises tool changes and tool path lengths and selects a suitable nibbling tool for non-standard features. This information is then converted into source code for the Wiedemann NC punch press used by Otis engineers. The system also has a nesting function, but makes no attempt to automate the planning of bending operations.

Shin et al (1990) have carried out further work in this domain introducing the additional complexities of a large variety of hole types, the necessity to change a tool during punching and the possibility of certain holes having to be formed before others for technical reasons. The system also includes a user defined maximum number of strikes allowed on one path to save computational time. Svestka (1990) introduces a new heuristic specifically aimed at more complex problems. With problems involving 30 or more hits this system out performs all existing heuristics.

The information required, from a nesting package, by the more complicated cutting processes can easily be derived from a layout designed for the sophisticated NC cutting methods. Thus any nesting system which is capable of supporting the most advanced NC cutting tools is also capable of supporting the simpler cutting methods. However, this is an inefficient planning route. It would be more efficient to select the level of output detail required which would then determine the nesting method. Alternatively, a nesting system could be used which initially generates a simple layout plan which can then be enhanced to the appropriate output level for the cutting method. This approach favours part simplification which reduces the complexity of the nesting problem.

1.8 Simplification of Parts

To save computational complexity and memory requirements many automated two-dimensional nesting systems use simplifications of the components rather than their true profiles. Although simplifying the parts to be nested may require considerable processing time, this is generally very low in comparison to the additional time required to manipulate unsimplified parts in the nesting system. An example part is shown in Figure 1.13. Five general levels of simplification have been applied to this example part which are shown in Figure 1.14 to Figure 1.18. Each figure also shows the particular nesting restriction incurred at each level of simplification. It must be noted that some of the forms of nesting shown in Figures 1.14 to 1.18 are beyond the ability of any current automated nesting system to create layouts of multiple parts.

The part in Figure 1.14 has had all its internal features removed. Generally internal features are too small for parts to be placed within them, however occasionally it can be done, as in the example shown. Figure 1.15 shows the part's 'convex hull', this is the shape formed if an imaginary elastic band were placed around the part's perimeter. This simplification prevents the concave sections of a part's perimeter being filled by convex sections of a neighbouring part. A straight line envelope to enclose the part is shown in Figure 1.16. This is effectively the convex hull of the part with the arcs represented by straight line sections. This removes the possibility of the arcs matching features in other parts. Figure 1.17 shows an enclosing rectangle constructed of rectangular sections. This limits envelope feature matches to only rectangular shapes. This form could also be produced by a single rectangular envelope with rectangular spaces within it. The simplest representation for a part is the smallest enclosing rectangle, as shown in Figure 1.18. No parts or sections of neighbouring parts can lie within this area. However, this simple format allows the component to be represented in space by just four values; the dimensions of the rectangle and the co-ordinates of its bottom left corner.

Note. **In Figure 1.14 to 1.18 the level of simplification is shown on the left hand side. The type of nesting which this stage of simplification has prevented over the previous stage of simplification is shown on the right hand side.**

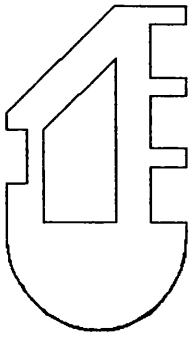


Figure 1.13.
Example Part.

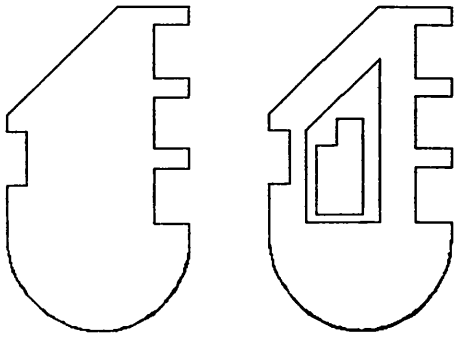


Figure 1.14
Part with internal features removed (see note at top).

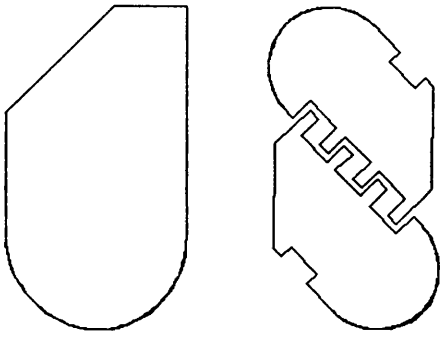


Figure 1.15.
Convex Hull of the part (see note at top).

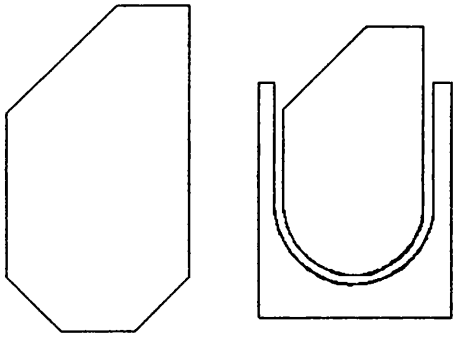


Figure 1.16.
Straight line envelope around the part (see note at top).

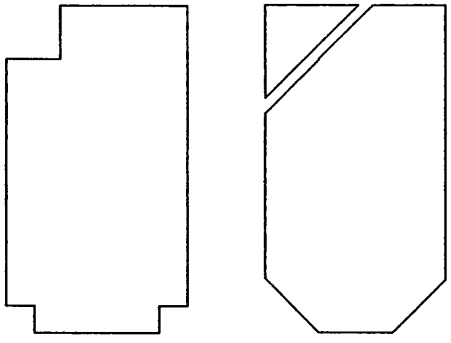


Figure 1.17.
Rectangular construct envelope around the part (see note at top).

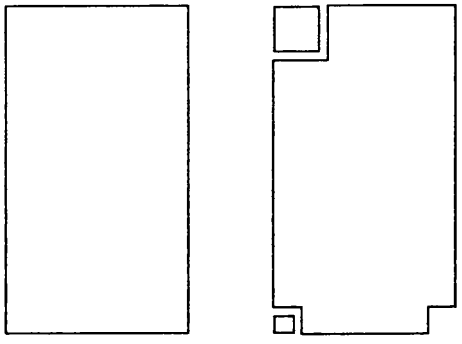


Figure 1.18.
Rectangular envelope around the part (see note at top).

1.9 Types of Layout Waste

When unsimplified parts are placed onto a sheet of stock material there are four discrete categories of waste, shown in Figure 1.19. This assumes that no other parts are small enough to be accommodated in the internal and inter-part waste areas.

(1) **Clamping Area Waste.** This is a relatively small area and in some situations it may be possible to utilise it by moving the sheet during cutting. Generally this area is unused as often the clamping method damages the surface, rendering it unsuitable for production.

(2) **Bridge Gap Waste.** This is the spacing required between parts due to the cutting method. It is normally accounted for by placing a perimeter boundary of the required width around the part. Punching processes often require a gap between parts to preserve local sheet stability during material removal. Laser and flame cutting processes require a gap to allow for metal removal and possible post-processing of component edges. This waste can not be avoided using current cutting processes.

(3) **Internal Feature Waste.** This can occasionally be reduced by nesting a small component within a hole in a larger component. If these features are considered as nesting waste a large amount of waste which is beyond the control of the nesting system is included. This would result in an efficiency statistic which does not directly reflect the performance of the nesting system. If internal features are ignored in the calculation of nesting waste, but parts are then placed within them, these parts effectively occupy zero sheet area in the calculation of nesting waste.

(4) **Inter-part Waste.** This is the material which remains between parts due to their awkward shapes preventing perfect part interlocking and a waste free layout. This is often the only waste considered when evaluating a nesting system. Quoted nesting efficiencies rarely refer to total levels of waste ie. material in / material out. With all but the simplest parts lists it is impossible to identify the optimum layout, therefore an absolute nesting efficiency cannot be quoted unless all waste is taken into account.

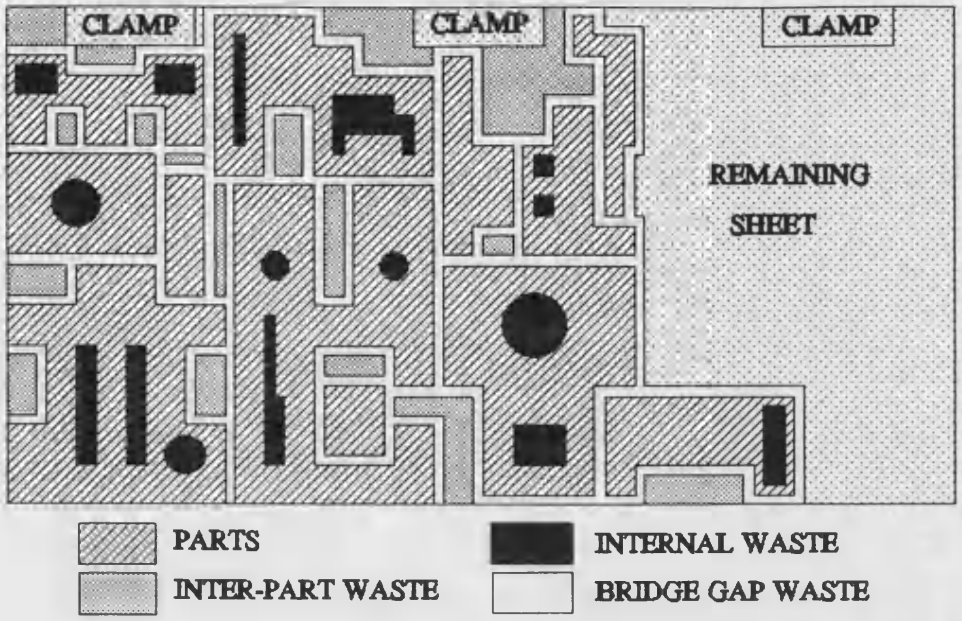


Figure 1.19.
Types of waste with unsimplified parts.

If the perimeter of a part is simplified or the part is enclosed within an envelope the ' Inter-Part Waste' is divided into the two following types.

(A) **Envelope Waste** incurred by enclosing the part. This is directly associated with the part and cannot be altered during the subsequent nesting process. The bridge gap is usually included in the envelope dimensions.

(B) **Inter-Envelope Waste** between all the enclosing envelopes. This is the only type of waste which is affected by the nesting process.

A layout of parts enclosed in rectangular envelopes, which includes these types of waste, is shown in Figure 1.20. A single part can be enveloped by a large range of shapes which form repeating uniform patterns (tessellation), as shown in Figure 1.21. In this situation all the Inter-Envelope Waste will occur at the sheet perimeter due to a mismatch between the edges of the tessellating pattern and the sheet.

So far the waste looked at has been in terms of material 'written off' by the process constraints or the nesting method. However, the true waste may be greater than this as some of the remaining material may be too small or too narrow for parts to be

placed on it. Material which is large enough to accommodate only a few parts may also be discarded if it is not economic to retain it or set up the machine for such a small number of parts. In a real production environment the largest rectangular section of the remaining sheet is often the only part which is retained, and this may rely on the section exceeding a minimum area. The actual limits for this decision will depend on the costs of the storage, the machine set-ups and the value of the material.

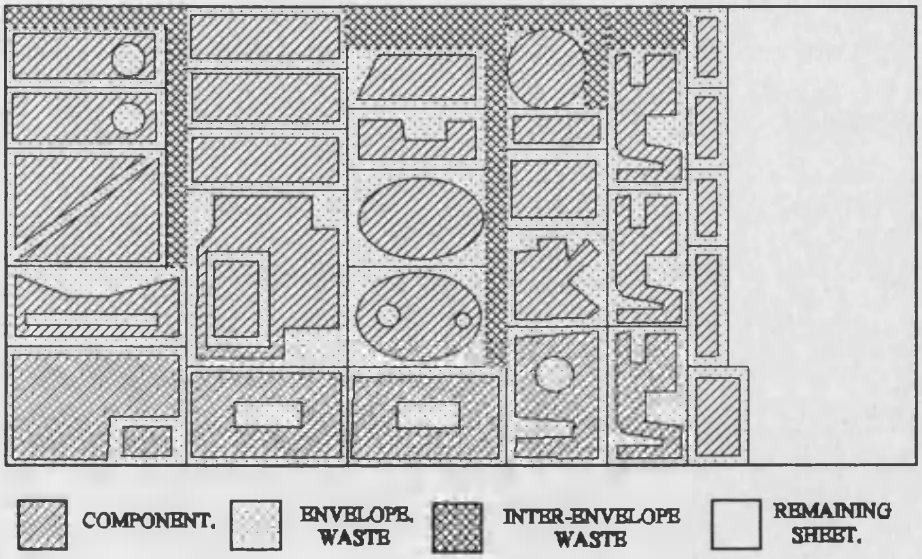


Figure 1.20.
Waste associated with parts enclosed within rectangular envelopes.

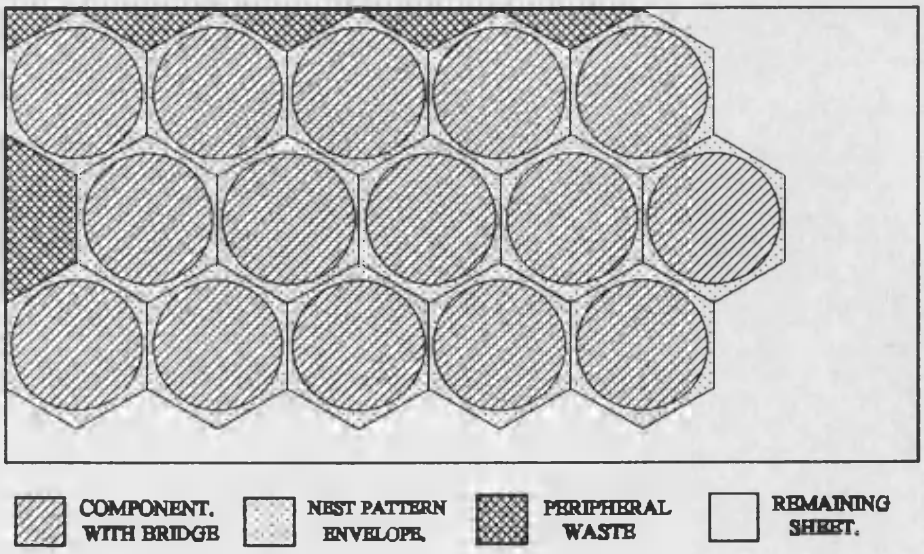


Figure 1.21.
Waste associated with a tessellating layout.

Chapter 2

Existing Nesting Systems

In this chapter the existing systems and methods of computer aided nesting are reviewed. A number of other areas are also reviewed which employ methods which may be adapted to tackle the 2-D nesting problem.

2.1 Cutting Stock Problems

This section aims to summarise and evaluate the published work carried out on the topic of nesting 2-D (two dimensional) parts or shapes into a pre-defined space. This problem falls into the overall category of 'cutting stock problems'. The systems addressing this general problem vary in their specific purpose and the method by which they operate. In addition, 1-D (one dimensional) cutting stock problem systems and 3-D (three dimensional) pallet loading or container packing systems should be considered as methods which are possibly applicable to the 2-D environment.

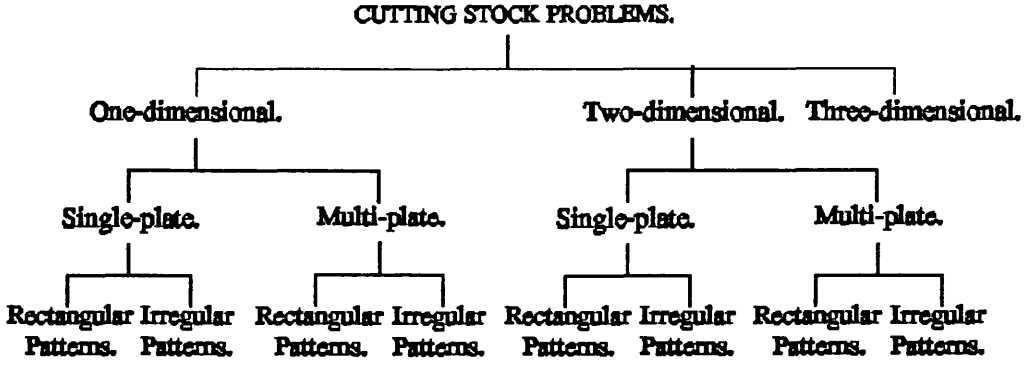


Figure 2.1.
Classification of Cutting Stock Problems by Dagli and Tatoglu.

It is necessary to classify the nesting systems in order to review them in an organised and meaningful way. Dagli and Tatoglu (1987) have suggested a classification for all cutting stock problems (Figure 2.1). However this system seems flawed as a one dimensional single 'plate' (length) problem can only exist in terms of finding the optimum stock size of material for a particular application. Also, the classifications below this level are in fact geometrically impossible. Almost all of the types of systems reviewed deal with single-plate and multi-plate problems in the same manner. This type of classification does not seem appropriate for the segregation of cutting stock problems encountered in this work. Thus a new classification system is suggested and shown in Figure 2.2. All systems encountered for the one dimensional environment which have been adapted to the two dimensional problem can be applied to the rectangular part nesting problem and are referenced in this section.

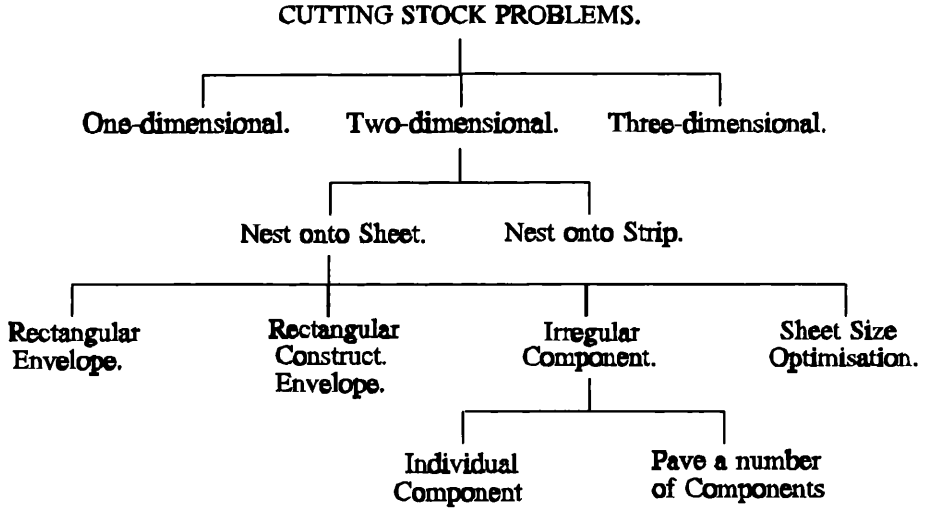


Figure 2.2.
New Classification of Cutting Stock Problems.

There are a number of interactive nesting systems which allow easier layout generation by a human planner due to fast part manipulation, bridge gap creation and layout efficiency calculation. As these systems do not include positioning rules they will only be covered briefly. Catastini (1976) developed one of the earliest interactive nesting systems. Ikeda (1979) introduced 'The Pansy', an interactive parts nesting system which was developed in Japan. The parts are roughly positioned using a joy

stick and then exactly positioned by specifying the distances from neighbouring parts. Sperling (1979) describes a similar system which was developed in Sweden. Some systems automatically generate a 'first attempt' layout which can then be further improved using an interactive system [Arndt (1979)]. Albano (1977) and Israni and Sanders (1985) also describe such systems, however the subsequent part manipulation is limited to a small number of options. Böhme and Graham (1979) discuss practical nesting experiences of nesting in the ship building environment and draw the conclusions that the optimum system is to combine automatic initial part placement with interactive layout improvement by the user. Hodgson (1982) has developed an interactive 3-D layout system for pallet loading.

2.2 Rectangular Envelope Nesting

All two dimensional cutting stock solutions developed from linear programming methods deal purely with rectangles, as do many other algorithms suggested. This requires all the parts in a real system to be placed into rectangular enclosures or envelopes before the system can deal with them. This obviously assumes and accepts a certain level of waste before the algorithm has positioned the parts. However this can be justified if the simplification of the part allows an algorithm to pack these envelopes very densely. If the parts have a basically rectangular form, as a considerable proportion of sheet metal parts do, the losses from enveloping will be low. Otherwise the losses tend to be considerable, although such non-rectangular shapes will typically give high losses with any of the current methods of nesting. These losses can often be reduced by clustering more than one part into a single rectangular envelope, which can then be nested.

The most basic 2-D cutting stock systems are developments of solutions for 1-D cutting stock problems. These 1-D solutions have generally been derived through Linear Programming (LP) techniques. Iterative LP solutions to 1-D and 2-D packing problems are described by Gilmore and Gomory (1961) and (1963), and Stainton (1977). The strength of the linear programming approach is the potential for mathematical performance analysis; within set constraints the solution can be proved to be optimal. However, for a large parts list, such an exhaustive search can prove too time consuming.

Herz (1972) proposes a recursive algorithm based on the iterative algorithms of Gilmore and Gomery. This operates by dividing the stock sheet by potential guillotine cuts and testing whether the resultant sheet segments can accommodate the parts which are yet to be placed. The algorithm requires 20% less computation time than that of Gilmore and Gomery to generate a layout with the same level of waste. The algorithm suffers from assuming that the parts list is sufficient to fill the whole sheet. Johnson et al (1974) introduce two quick 'non-optimal' 1-D packing algorithms and give proofs for their worst case performance bounds.

Page (1975) tackles the problem of selecting the optimum range of bar lengths for a firm to hold in stock, from which smaller section pieces can be cut, given the range of sections required. However the solution method, obtained by constrained dynamic programming, can not be adapted to create sheet layouts. Dyckoff (1981) describes a Linear Programming (LP) technique which was applied to 2-D orthogonal cloth cutting for a German textile company. However the constraints imposed effectively limited the problem to a single dimension. This was reported to be acceptable, as the large number of residual strips of cloth could be stored for future orders. In the sheet metal environment, a complicated stock management system would be required to manage the large number of off-cuts and allocate them to parts in the parts list. This would also involve the company carrying high levels of raw material. Chambers and Dyson (1976) and Beasley (1985) describe algorithms to determine the optimum stock sheet sizes to produce, such that they can be then guillotine cut to exact customer specified sizes. Tokuyama and Ueno (1985) describe a near optimal system to solve this problem and its application in a sheet steel mill.

Gilmore and Gomery (1965) applied their LP work, developed to solve 1-D cutting stock problems, to the 2-D and 3-D situation. Haims and Freeman (1970) and Christophides and Whitlock (1977) have tackled the 2-D multi-variable problem by converting it into a multi-stage problem, by imposing considerable constraints, which they solve using dynamic programming. A comprehensive survey of early LP approaches to packing problems is given by Hinxman (1980).

Much work has been carried out to prove the worst case performance of simple nesting algorithms. Kleitman and Krieger (1970) proved that any number of square parts, whose areas sum to unity, can all be located within a rectangle of sides 1 by $\sqrt{3}$. The area of this rectangle is considerably larger than that required by any practical system when applied to such a simple problem. Thus mathematically proven worst case bounds for these simple situations are only of academic interest, and have little relevance in the search for practical part nesting systems. Gardener (1979) reviews a large amount of theoretical work proving the minimum square area which will enclose various numbers of identical squares. This work cannot be applied to sheet

layout generation as enclosing parts within squares (rather than rectangles) would often incur huge levels of waste, only one square size can be used and no deterministic method is given to select the pattern of squares to be used.

Adamowicz and Albano (1976b) recognised that planners in industry frequently form strips of parts to aid subsequent nesting and have developed an orthogonal nesting system which operates in a similar manner. The parts are defined as being principally orientated i.e. the part must be horizontal or vertical in relation to the sheet. The paper also defines four types of strip formation, shown in Figure 2.3.

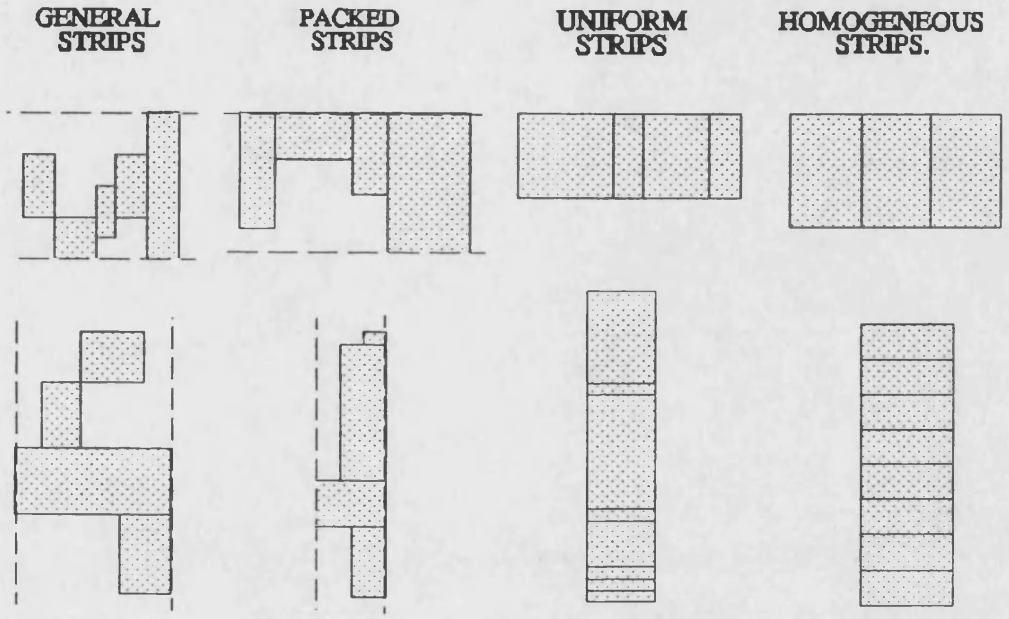


Figure 2.3.
Strip Type Definitions by Adamowicz and Albano (1976b).

The Adamowicz and Albano system operates by forming 'homogeneous' strips from single part types and uses them if their length is less than the sheet length and greater than a waste acceptance limit. If the available strip lengths are too small, the sheet is divided into sub-sections into which the strips are nested. This is done by provisionally nesting the largest strip and splitting the remaining sheet length at the end of this strip. If strips are available which fit into the areas created, this sheet division is accepted, otherwise the next longest strip is tried. Any strips of parts are longer than the sheet are divided into the longest sections which will fit. The system

also allows free rectangular spaces at the ends or sides of nested strips (numbered 1 to 5 in Figure 2.4) to be subsequently filled.

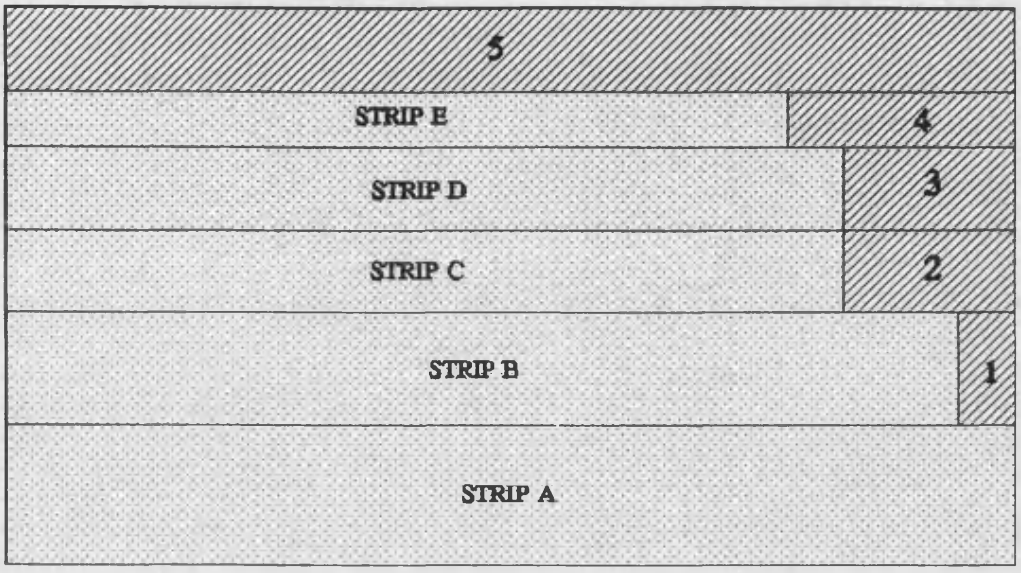


Figure 2.4.
Strip Layout by Adamowicz and Albano.
(Numbers denote free rectangular spaces to be filled)

Albano and Orsini (1979b) have developed an improved version of the system described by Adamowicz and Albano, with the constraint of forming purely uniform strips removed. The strips are formed of quasi-uniform strips i.e. any parts within a specified width limit. The parts are still taken in a random order, part rotation is not permitted and the guillotine cut constraint cannot be removed. The system gives excellent results with a large parts list of similar parts which are small relative to the sheet size. However, a small list of large diverse parts would not be nested efficiently.

De Cani (1978) introduces the concept of a cutting pattern of orthogonal parts which are not constrained to the principle orientations of the sheet. This approach would increase the computational complexity of an automatic part placement system and invariably reduces the efficiency of the layout generated. The approach permits the positioning of parts which are too big for a sheet when constrained to principle orientations. i.e. a 21 by 1 part onto a 20 by 20 sheet across the diagonal. This

eventuality is rare and could be dealt with manually or a larger stock sheet may be available. Allowing parts to stray from the sheet's principle orientations within an automated nesting system is considered to add significant complexity for negligible benefit.

Many 2-D orthogonal nesting systems deal with the academic problem of positioning rectangular shapes into an open ended bin. Baker et al (1980) investigate the performance of the bottom-up, left-justified (or BL) algorithm; which attempts to minimise the height of the packing of orthogonal shapes into an open ended bin. This algorithm can easily be adapted to nesting parts onto sheets. The parts are taken in a pre-defined order and located as far down the bin and then as far to the left as possible (Figures 2.5 & 2.6).

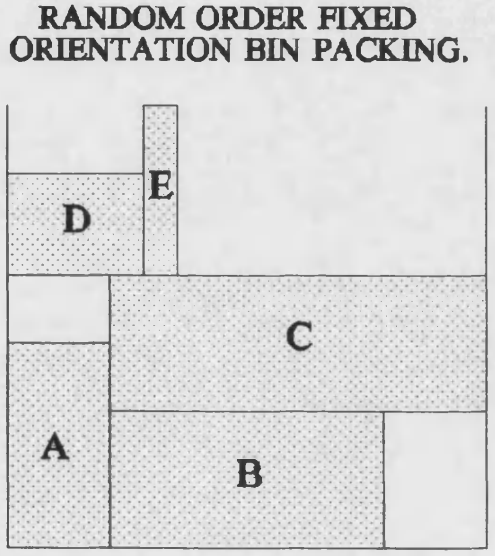


Figure 2.5.
Random bottom left packing.

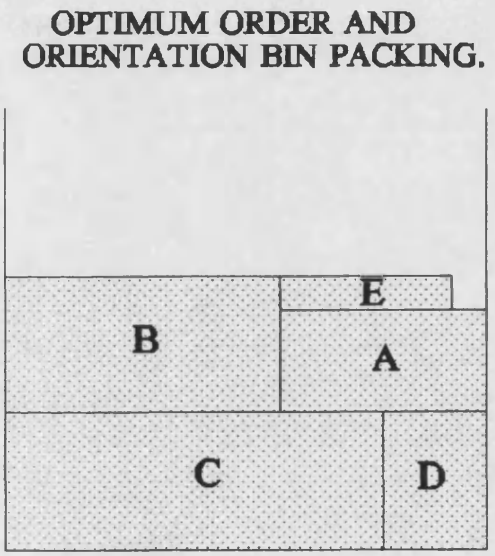


Figure 2.6.
Optimum bottom left packing.

In Figure 2.5 the parts have been positioned in alphabetical order and in their original orientation. The re-orientation of shapes A, D and E, and altering the positioning sequence to C, D, B, A and E yields a considerably better result, shown in Figure 2.6. Thus the effectiveness of this algorithm is reliant on the sequence of the parts, which can only be optimised by trial and error. Allowing the freedom to rotate a part can

only improve the nesting system as it will increase the range of positions which a part can adopt. In practice parts may have a pre-defined orientation, for instance to keep folds perpendicular to the grain to maintain their strength, therefore rotation may not be permissable. With rectangular envelopes or modules, only primary orientations (90° rotation) should be considered, as others will interfere with the overall nesting pattern. The parts can be sequenced to prevent the worst case performance occurring, by using the parts in decreasing height or decreasing width order, as in Figure 2.7 and Figure 2.8. These sequences rarely approach the performance of the optimum, but are quick to establish.

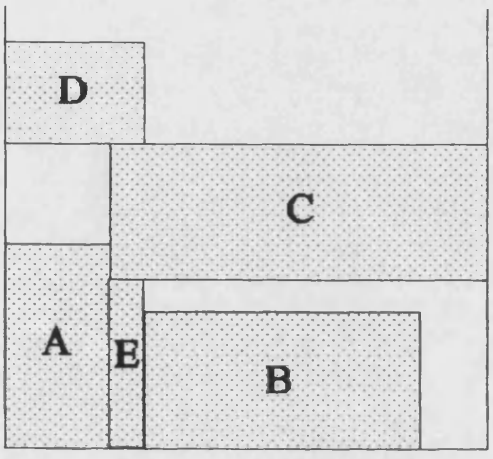


Figure 2.7.
Decreasing height sorted
BL packing (A,E,B,C,D).

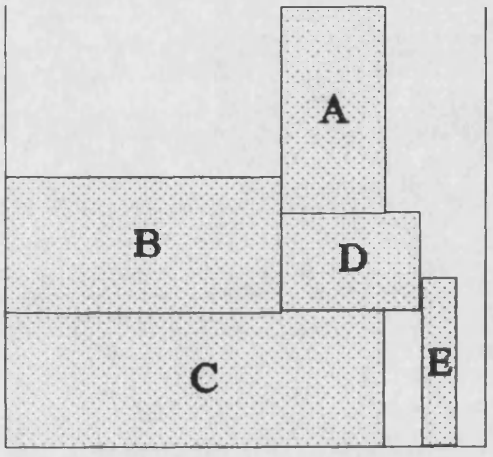


Figure 2.8.
Decreasing width sorted
BL packing (C,B,D,A,E).

Bottom Left location algorithms do not always generate patterns which can be produced using guillotine cuts. As discussed, guillotine cuts are those which span the entire sheet when they are cut. This is necessary when using traditional guillotine shears, however the advent of rectangular shears and laser cutting mean this constraint is often unnecessary. A versatile system would give the option of creating a guillotine cut constrained nesting pattern. Alternatively this could be a first stage, to be improved to a non-guillotine cut constrained layout if permissable.

Coffman et al (1980) describe four bin packing algorithms, the first of which is a Bottom-Left location algorithm as already described. Figures 2.9 and 2.10 show the Next-Fit Decreasing-Height and First-Fit Decreasing-Height algorithms (in both diagrams the parts have been placed in alphabetical order). The former locates each part, in height order, in rows. When the next part in the list is too large to fit in the current row, a new row is started. The latter algorithm operates in a similar manner, however each remaining part in the parts list is checked for a fit into the waste areas remaining on the ends of the previous rows. Neither algorithm allows re-orientation of the parts. With a larger parts list there would be less height difference between neighbouring rectangles which would lead to lower proportional waste.

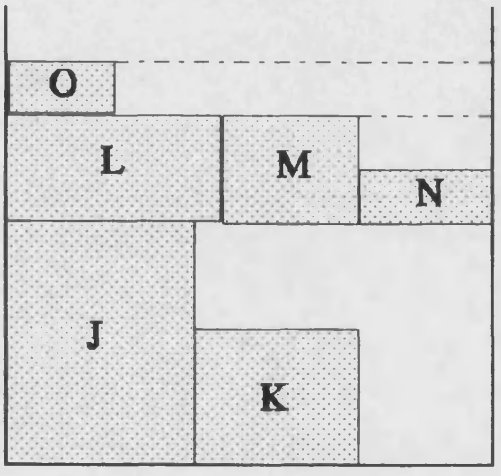


Figure 2.9.
The Next Fit Decreasing Height algorithm.

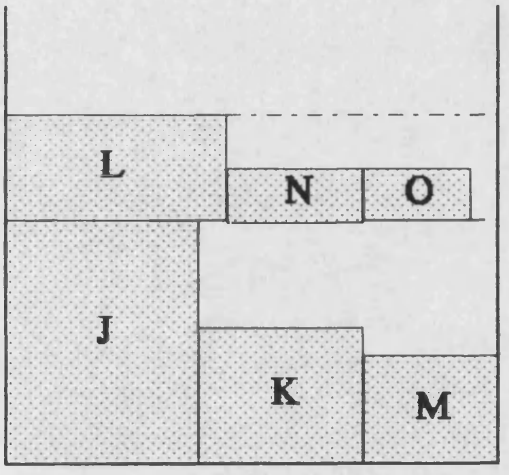


Figure 2.10
The First Fit Decreasing Height algorithm.

The final algorithm described by Coffman et al is their Split-Fit algorithm shown in Figure 2.11. This system splits the parts list into two groups, above and below a certain height. This allows a rectangle to be inserted above the lower height group into which further parts can be placed. This works well for a large parts list of strips, but would be less effective for a more varied group of orthogonal shapes.

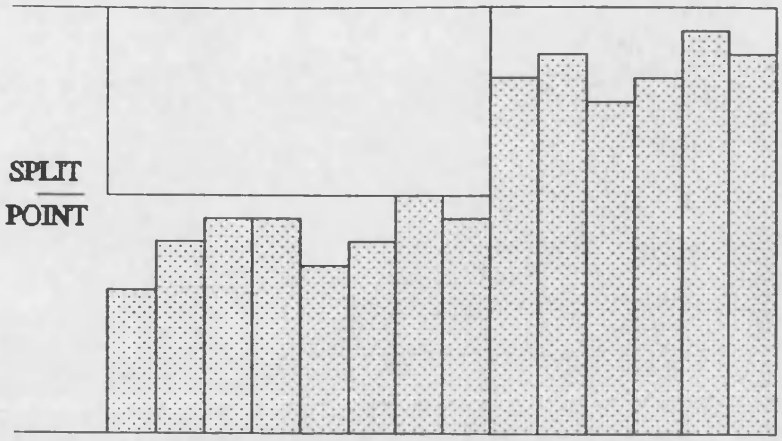


Figure 2.11
Layout generated by the Split-Fit Algorithm by Coffman et al.

Golan (1981) gives worst case performance bounds for two 2-D packing algorithms. For the first algorithm the parts are sorted into decreasing width order and the widest parts are placed across the base of the sheet and as far left as possible. When the area at the right end of the parts is sufficiently wide to accommodate some of the smallest (in width) remaining parts, a vertical split is made in the sheet to create a second area for part placement. As the parts further decrease in width more splits are made. The second algorithm generates better layouts by selecting groups of parts of similar width and then placing together two groups whose combined width is close to that of the sheet. A worst case performance limit for the second algorithm is given in terms of the maximum sheet area being $^4/_3$ of the total part area ie. 75% material utilisation. This value must refer to constrained part inputs as a part list could easily be designed which would exceed this waste level.

Wang (1983) describes two similar combinatorial methods for nesting rectangular parts on a specified sheet according to the guillotine cut constraint. Part rotation is not permitted. Both methods take the first part from an unordered list and sequentially try all the remaining parts above it and to the right of it. For each part and position the minimum envelope to enclose the pair is applied and the waste within it calculated. The first part with an associated waste level below a preset threshold is placed. This process is repeated, with the existing layout being treated in the same

way as the first part, until all the parts are placed. The two methods differ in the application of the waste threshold. Both methods give good layout efficiencies considering the constraints imposed.

Albano (1977) developed a system which creates a layout using a simple rectangular envelope placement algorithm which the user can then improve using an interactive system. Israni and Sanders (1985) recognised that the layouts of 'guillotine cut constrained' nesting systems could be further improved by removing the constraint. Altering an algorithm to remove this constraint will affect the fundamental way in which it operates. Thus the authors advocate a limited amount of human intervention to adapt the pattern produced by the algorithm. To quantify the improvements obtained through this methodology the authors constructed a range of experiments to test the performance of six individual algorithms, two of which included human intervention. The algorithms tested were as follows :-

(1) DLPER - Decreasing length perpendicular strip placement (see Figure 2.12). This algorithm, developed by Israni and Sanders (1985), places parts along the left side and the base of the rectangle or sheet. The parts are taken in decreasing length order, no further ordering is made for parts with the same length. In the example given length is in the X-axis. The placement initiates in the bottom left corner and alternates on each part from positioning on the base and the side. Nesting in both horizontal and vertical axes simultaneously removes some of the benefits of the length ordering, intended to produce vertical strips of parts with similar widths (X dimensions).
(2) DHPER - Decreasing height perpendicular strip placement. This operates in an identical manner to the DLPER algorithm, but the parts are located in ordered of decreasing height.
(3) Bottom-Left First-Fit Decreasing Width. This uses the standard bottom left location algorithm, described earlier, with parts ordered by decreasing width.
(4) First Fit Decreasing Height. This algorithm, developed by Coffman et al (1980), has been described earlier.
(5) DLPER with human intervention.
(6) DHPER with human intervention.

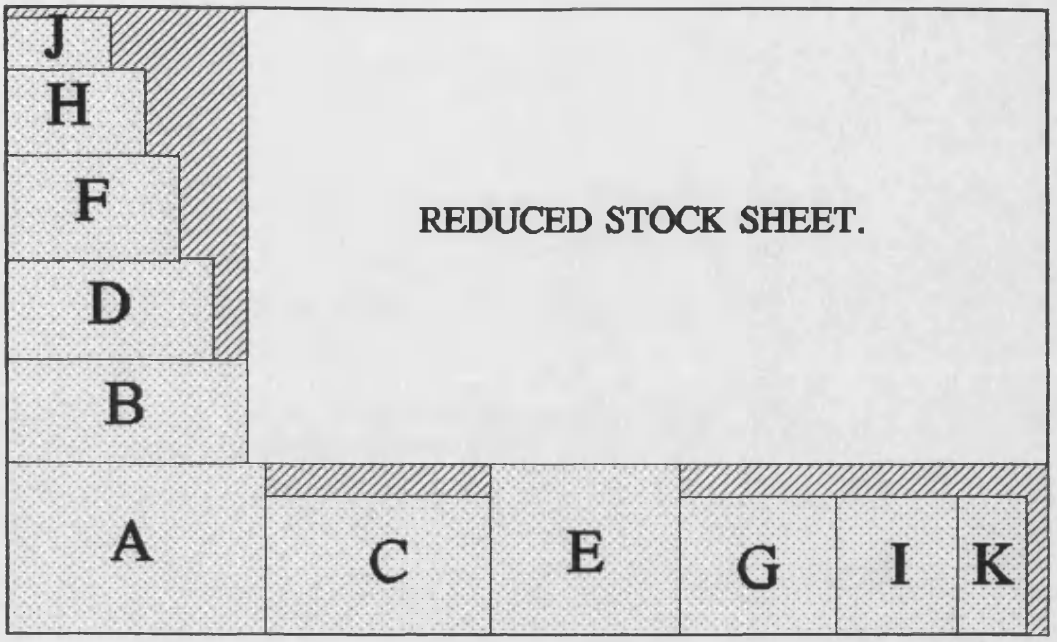


Figure 2.12
DLPER Heuristic by Israni and Sanders.

The systems featuring human intervention out performed the others, with the accepted disadvantage of not being fully automated. Human intervention produced the greatest improvements when dealing with a few large parts on a sheet. This is due to the unwieldy nature of large parts for automated systems and human planners excellent spatial reasoning with a limited number of parts. Conversely limited improvements were made with sheets containing a large number of small parts, as this problem is more suited to automated systems and therefore a good layout will be produced. In addition the large number of parts makes their re-positioning time consuming for a human planner. Further details on this system are given by Israni and Sanders (1982).

Bengtsson (1982) describes a heuristic approach to optimise the part sequence used for a Bottom-Left algorithm. The system maintains a record of the best solution so far and the ability of a sequence to improve on the best solution is checked throughout the placing of the parts. The remaining parts in the parts list are assumed to be nested with no waste to give a projected minimum final area. If this exceeds the best performance so far, this nesting sequence is abandoned and a new sequence tried.

When the parts list has been located onto a number of sheets, the parts on the least efficiently nested sheet are placed back into a part pool, along with the parts of the last sheet, which in most cases will only be part filled. This reduced part range is then re-nested to produce another full sheet and a part filled sheet. The process is repeated until no further improvement is seen in the re-nested sheets. This system has the potential to be time consuming for small improvements in the overall nesting efficiency and it tends to leave all the awkward parts for the last sheet, resulting in it being poorly nested. As the last half-filled sheet is not considered when evaluating the overall nesting efficiency this problem is removed from the results. The principle of returning the parts of poorly nested sheets to the parts list would possibly be more effective if another nesting algorithm or sheet size was then tried.

Nee et al (1986) have developed a nesting system for parts enclosed within rectangular elements which does not guarantee a solution which conforms to the guillotine cut constraint. Any parts described by lines and circular arc segments can be placed by the system. The arcs are subsequently simplified by chords of a user defined length. If a batch of one part is required this simplification allows a pairwise cluster of parts to be created. If this clustering offers a material saving in excess of 20% when placed in a rectangular envelope it is used. The 20% acceptance threshold is to account for the unwieldy nature of the larger envelope during subsequent placement. If a single part is used the arc approximation is superfluous due to the subsequent enclosure within a rectangle. The envelopes are taken in descending area order and the largest is placed at the bottom left of the sheet. Pivot points are created at concave line intersections and the next part is tried in each orientation at each pivot point (Figure 2.13). The part giving the lowest total occupied area is selected. This process is repeated for all parts.

This system also tackles the problem of an irregular sheet shape such as a leather hide. The sheet is approximated with straight lines, each of which becomes the diagonal of a rectangle (Figure 2.14). Only the area bounded by these rectangles can be nested, accepting considerable peripheral waste. The system also allows sheet 'bad areas' to be enclosed by a rectangle and excluded from part placement.

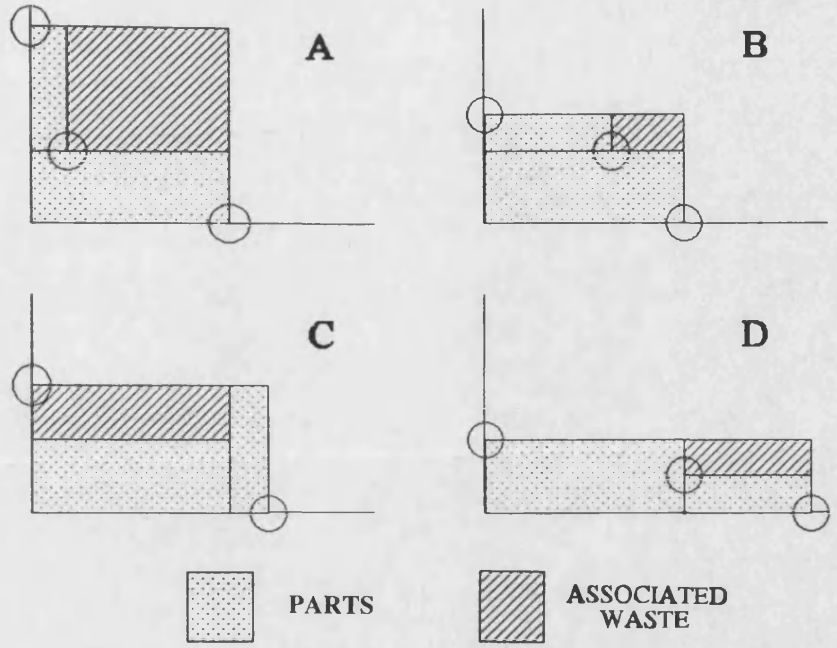


Figure 2.13.
The locations for the second part using Nee's algorithm (The circles represent the pivot points for the next part's location).

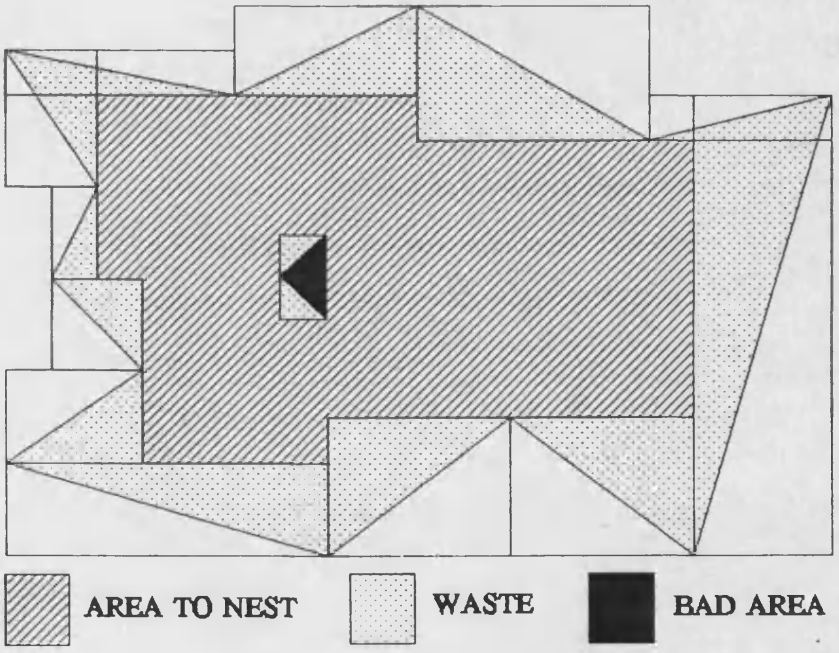


Figure 2.14.
Nee's system for irregular sheets.

Flemming (1986) describes a commercial sheet metal planning system Q-NEST by Counting House Computer Systems. However, no details are given of the algorithm which is employed to generate layouts. A nesting efficiency of 70-90% is quoted.

2.3 Rectangular Construct Nesting

Layouts generated by this method do not conform to the 'guillotine cut' constraint. Roberts (1984) developed a heuristic system which could nest 'L' shaped envelopes as well as rectangles, however no detailed information is given on the rules employed.

Qu and Sanders (1987) identified that many parts encountered in industry which do not conform to a single rectangular form can be accurately represented by a construct of a small number of non-overlapping rectangular envelopes. Thus improving the component representation. The accuracy of the system can be tuned to individual needs by altering the size of the minimum rectangular envelope. The rectangles which are linked in a construct envelope must maintain their positions relative to each other increasing system complexity. Thus, although this system has the potential to out perform other systems which use a single rectangle to envelope the component, the solution requires considerably more computation. The parts, ordered by decreasing height, are located using a Bottom-Left algorithm. If a placed part has a construct form with a stepped profile a part with a matching feature space is searched for. Both this part and the next height sorted part are provisionally nested (Figure 2.15). The part which results in the least waste is selected. An additional feature is that the system creates a layout in both sheet axes and then selects the better.

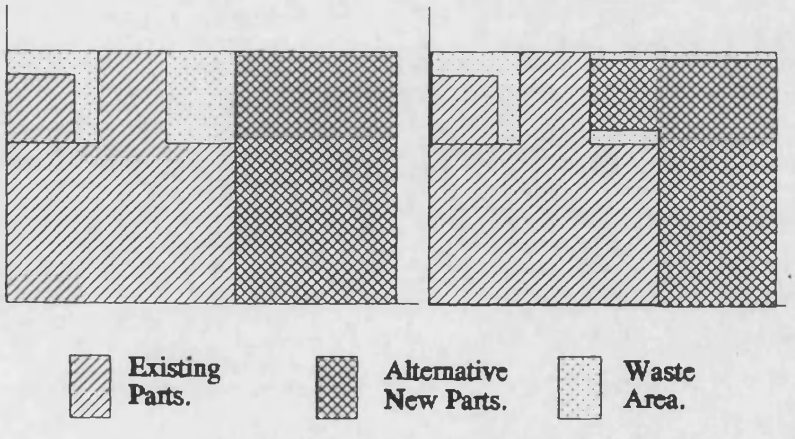


Figure 2.15.
Choices of 'Next Part' with Qu and Sanders system
(The right hand part would be chosen).

2.4 Irregular Part Nesting

Adamowicz and Albano (1976a) have developed a system to nest irregular parts into rectangular enclosures. This is intended to position awkward irregular parts into rectangular envelopes to be subsequently nested by another system. The system could be used to nest large irregular parts directly onto stock sheets, however the processing time would be considerable. Also, a low waste level could only be guaranteed if a good part sequence could be determined. The system has three levels of complexity, which are as follows:-

(1) A single part into a rectangular enclosure.
(2) A pair of identical parts into a rectangular enclosure.
(3) Either of the above with any spaces in the enclosure subsequently filled with parts.

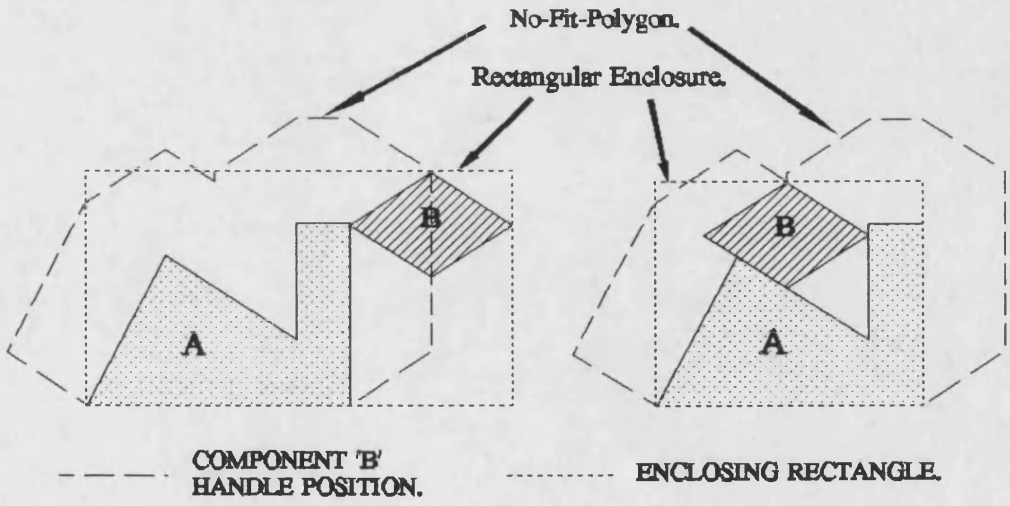


Figure 2.16.
Adamowicz and Albano's No-fit Polygon method of clustering.

The system with the third level of complexity creates the part clusters with the least waste. This system employs a 'No-fit Polygon' approach shown in Figure 2.16. The part (or parts) already positioned remain stationary while the part to be nested is moved around its free perimeter in a fixed orientation. The 'no-fit polygon' is the trace of this part's datum point or handle as it is moved around this path, in the case of part

'B' in the figure this is its top corner. The 'no-fit polygon' can then be analyzed to reveal the nodal point which allows the smallest enclosing rectangle. The system can also test for non-nodal positions which improve on the nodal performance. The parts selected to fill any spaces in the nesting pattern, with the system operating at complexity level three, are the largest parts fulfilling the following criteria :-

(1) Part area less than the waste area to fill (no guarantee that the part will fit).
(2) Parts whose 'number off' is greater than or equal to the number of waste areas to fill.
(3) Parts with a set of spans (dimensions) within those of the waste area to fill.

Albano and Sapuppo (1980) developed the work of Adamowicz and Albano (1976a) to allow the total nesting of a sheet. The system is based on the exhaustive application of the 'No-fit Polygon' approach. However, to reduce overall computation time heuristic rules are used to truncate the search which removes the guarantee of optimality (Albano and Orsini (1979a) give a detailed analysis of such truncation methods for this type of application). The parts are also limited to being tried in their normal and 180° rotation orientations. A good layout is achieved, but no computation times are quoted. Schalla et al (1991) have developed an integrated CAD/CAM and production control system dedicated to sheet metal manufacture which employs the nesting algorithms developed by Albano and Sapuppo.

Sanders and Bronsoiler (1983) describe a new algorithm for nesting irregular parts. The part areas are compared to those of their 'Minimum Enclosing Rectangle' to evaluate their complexity. Parts are sequenced for nesting largest and least complex first. The parts are placed in their unsimplified state towards the bottom left of the sheet, but can be moved anywhere along the perimeter of the parts already positioned. The final position of each part is selected on the basis of the least addition in the total area occupied by placed parts. The parts tested were generated by a random system developed specifically for testing the nesting system, and all are simple straight line shapes. No layout efficiencies are given for the system.

Dagli and Tatoglu (1987) propose a two stage approach to part nesting. The levels of complexity of the parts are analyzed by comparing their areas to those of the smallest rectangles which would enclose them. The highest value attainable is one, implying the part is a rectangle. The parts with complexity values of one are positioned by mathematical programming. If sufficient parts share one dimension the problem is reduced to a single dimensional cutting stock problem. Otherwise a grid is constructed with the element size equal to the largest common divisor of parts and sheet. This maintains perfect part representation at the expense of great complexity as real parts will require a tiny grid element. A pre-set grid size, in which parts adopt the minimum grid elements to enclose them, could reduce this complexity with limited loss in accurate representation. For non-rectangular parts a second system employing a heuristic to carry out nesting is used. The nesting sequence used is operator selected from the options below :-

Maximum Area First.
Minimum Area First.
Maximum Irregularity (complexity) First.
Minimum Irregularity First.
Maximum Number of Sides First.
Minimum Number of Sides First.
Maximum Number of Arc-Shaped Sides First.
Minimum Number of Arc-Shaped Sides First.
Maximum Perimeter First.
Minimum Perimeter First.
First-Come First-Served.
User Priority First Served.

Each part is exhaustively positioned on the sheet until the position which yields the minimum area of all the parts nested so far is found. This packing does not encourage the sheet to be entirely filled, parts will initially adopt efficient positions in the central areas of the sheet, rather than filling the more difficult perimeter sections. Also, it is doubtful whether a planner could develop rules to determine a good sequence for

irregular parts without some trial and error on the particular parts list, which would increase processing time. The system produced a lowest scrap rate of 7.7% (Maximum Area First Rule) and a highest scrap rate of 29.7% (Minimum Area First Rule) compared to a quoted industry average of 20%. These results support the generally accepted strategy of nesting the largest parts first.

Yuzu et al (1987) describe an irregular parts nesting system developed for the cloth cutting industry. The system imitates the nesting procedure of human planners; that is to place the largest parts and then fill the areas between them with the smaller parts. Gaps are left between large parts for the smaller parts on the basis of experience and suitable gaps vary with the type of parts list entered. The human planners knowledge in this area has been captured within an expert system module. This examines the parts list prior to nesting and determines the split point between large and small parts in the list and the gaps which should be left between the large parts.

The large parts are then placed using a set of complex rules to determine sequence and position. A second module places the smaller parts into the remaining areas. Finally, the layout can be further improved by changes implemented by an automatic feedback module or by the user through an interactive module. An operation time of around 3 minutes is quoted for a normal application. Without feedback the system achieved around 17% waste compared with other automated systems which are quoted at 20% and hand-made layouts at around 12%. With automatic feedback this dropped to around 15% and with interactive feedback to around 12%. It must be noted that this system is considerably faster than preparing a layout by hand.

Tiow et al (1992) describe a system to interface with the popular AutoCAD system. The CAD drawings are converted into ASCII code and exported to a Fortran nesting program which carries out an exhaustive translational and rotational position search for each part in turn. The system nests the parts in the order specified by the user. The system is slow for a small number of parts due to the exhaustive searching, making it inappropriate for large parts lists.

Parry-Barwick (1995) has applied a system developed for feature recognition to the two dimensional layout problem. Parts are given additional dimensions to model their positions in space. The possible positions of a fixed orientation part on a sheet could be modelled in four dimensions, those of the sheet and a further two to represent the possible positions of the part. Figure 2.17 shows a pictorial representation of the part, extruded at 45°, to show the possible locations in one dimension. The free space on the sheet is shown on the X axis and the boundaries of this space are cast vertically to bound the part's possible locations. A further dimension would be required to model the part's orientation. If the parts, which can be rotated, are placed one at a time the model will require five dimensions, two for the layout so far and three for the current part. The location of more than one part can be carried out simultaneously, however each additional part will require three additional dimensions to be added to the model. For n parts being simultaneously located, the model will require 3n + 2 dimensions. The number of dimensions included vastly increase the processing time. Thus the parts are placed one at a time in a pre-determined sequence.

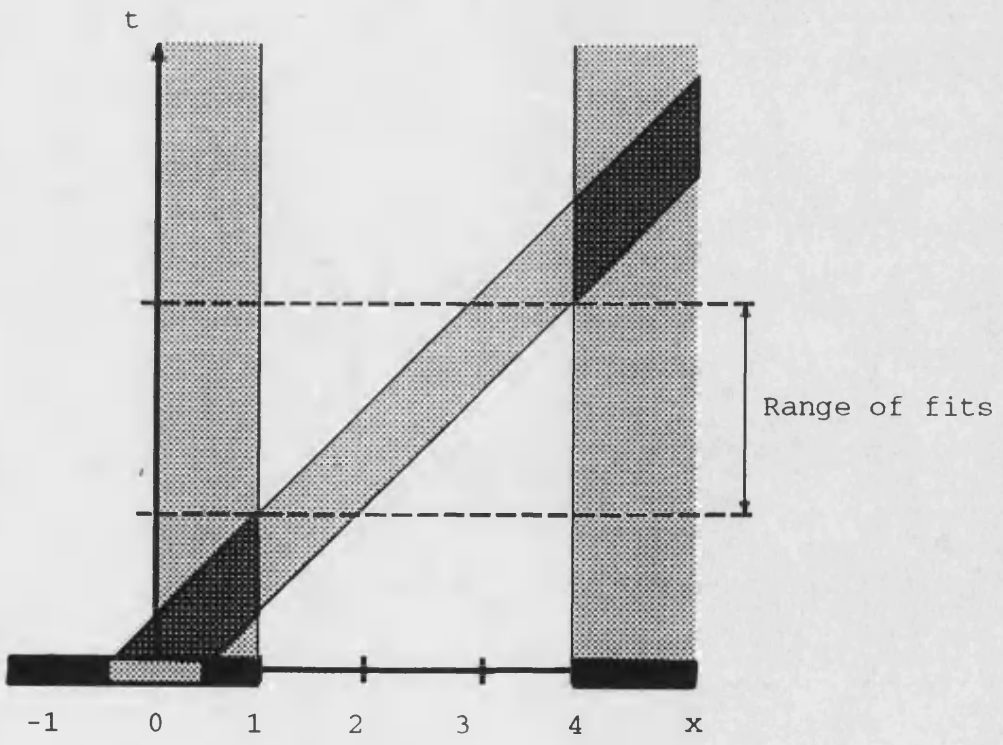


Figure 2.17.
Possible part locations in one dimension with Parry-Barwick's system.

The parts are sequenced by decreasing area and placed as close to the bottom left of the sheet as possible. The optimum part location is found via a process of 'binary spacial division'. The area of the sheet is split in half and each side tested for a boundary intersection. If no intersection is present the area is dismissed, otherwise the area is split and the process repeated. This allows the system to quickly focus on the relevant area without unnecessarily examining large free areas. If the binary spacial division is continued to a small division size, the boundary intersection (the optimum part location) will be pinpointed accurately. If the division is terminated at a coarse division the location will be inaccurate or a less good location identified.

The tests carried out were for the actual parts with complete detail or with their enclosing rectangular envelopes. Figure 2.18 shows a layout of 9 parts created by this system. This layout required 20 min. 40 sec. to create (using a Silicon Graphics machine with a 100MHz CPU) to create and occupied a rectangular area 8% smaller than the layout of equivalent rectangular enveloped parts, which required 2 min. 17 sec. In some tests the layout of rectangular envelopes occupied a smaller area than that of the actual parts.

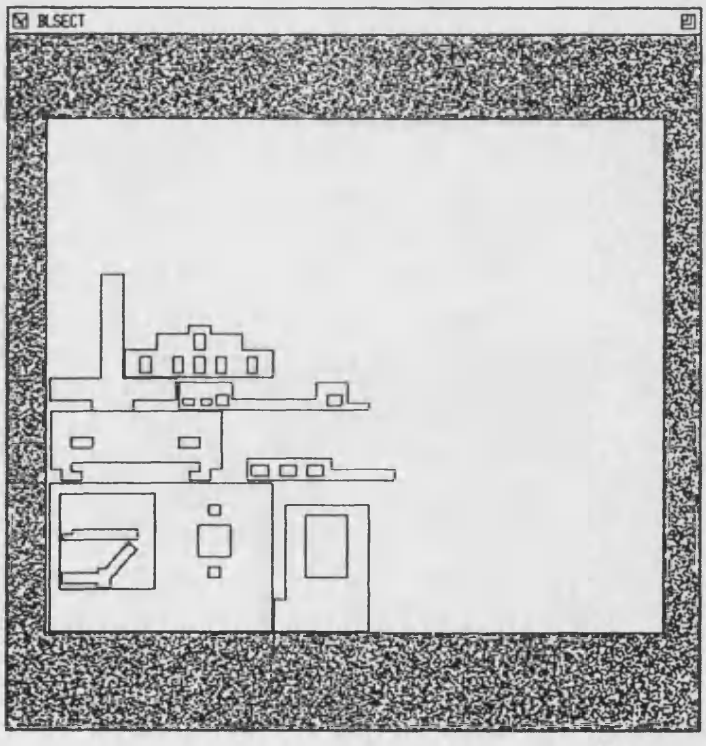


Figure 2.18
A BL (bottom left) layout of 9 parts.

The layouts of rectangular envelopes are not significantly worse than the layouts of the actual parts for the following three reasons.

(1). BL packing encourages parts to adopt a triangular pattern rather than an efficient rectangular group.
(2). The part placement sequence is often far from optimal.
(3). Many real parts have a generally rectangular form, so there is little disadvantage in using the rectangular envelope representation.

It must be noted that this system is not specifically directed at nesting and yet provides good layouts and is exceptionally good at placing parts within the 'internal waste' areas of other parts (Section 1.9). The system's processing time may be reduced by selective simplification of parts. This system could easily be adapted to create efficient clusters of highly irregular parts within rectangular envelopes, to be subsequently nested by another system.

Freeman and Shapira (1975) describe a system for finding the rectangle of minimum area which can enclose an irregular part to allow it to be nested by an algorithm which is constrained to nesting rectangular parts or envelopes. Dori and Ben-Bassat (1983) describe a method of simplifying an irregular part to an n sided convex polygon. This system allows the level of part simplification to be determined by setting the number of sides of the polygon which is used to represent the part.

2.5 Nesting Parts onto Strips

There are a number of documented systems which are designed to orientate and repeat the same part onto a suitable width strip for blanking. Some of the systems can also pair parts efficiently for simultaneous blanking from the strip. These systems address a different problem to those described so far. The large production volumes and the simplification of only considering a single part makes the exhaustive search methods required to find the optimal solution viable.

Nee (1984b) developed a system for pairing blanks enveloped by a polygon, where trial orientations are constrained to those of the polygons' sides. An n sided non-concave polygon has n possible orientations and an n sided polygon with i concave vertices has n-2i possible orientations. The algorithm translates one part relative to the other until its optimum position for the orientation is established. This is carried out for all part orientations and the best six results are stored. The strip widths required by each of the six best layouts are calculated and compared to a list of stock strip widths to allow the most efficient to be selected.

Nee (1984a) proposes two algorithms to determine the optimum pattern with which to blank a part from a strip of material. The first algorithm evaluates the material utilisation achieved with the part in numerous orientations, incrementing the part orientation in 2° steps. The second algorithm considers a pairwise part cluster, with one part rotated through 180°, suitable for a multiple station press. Both algorithms can incorporate limitations in lengths and widths of the strip from which the parts are to be blanked, and the orientations of the parts. A sheet can also be nested with one part by considering it as a set of parallel strips of the required width for the part. The layouts generated are good, however the second algorithm needed 20 minutes to nest a sheet with a part of reasonably complex shape.

Illiev et al (1989) have developed a total planning system for the design of the tools required to blank or wire cut a part from a sheet. The nesting element of this system tries the single part in 5° incremental orientations from 0° to 180°. The best solution

is chosen on the basis of the closeness of packing between sequential parts and the width of the strip required. The system can also create a pairwise cluster of a part and carry out similar calculations to establish the optimum orientation for simultaneous stamping. No material utilisation statistics are given for the system.

Prasad and Somasundaram (1991) have developed CASNS (Computer Aided Sheet Nesting System). The nesting module of CASNS consists of three layout formats: SPSR (single part, single row), SPMR (single part, multiple row) and MPSR (multiple part, single row). The SPSR and MPSR are intended for creating a strip layout and examples are given of paired parts as well as single in a repeated pattern. The SPMR system simply repeats rows generated in the SPSR format across a sheet. The computation times for all the layout formats are reduced by an optimising module which constrains the exhaustive search of possible solutions. This system is not intended for very low volumes or a large variety of parts on a single sheet.

Ismail and Hon (1992) have also developed a system for pairing identical parts to be simultaneously punched from strips. The part is overlaid onto a fine division grid and subsequently represented by the elements of the grid which it covers. This method of part simplification is similar to representing a part by a construct of rectangles discussed earlier (Qu and Sanders 1987), but gives finer part representation. Two of the simplified shapes are used to represent the pair of parts; one shape remains stationary while the other is evaluated in various grid positions around it to find the pairs optimum position. The grid representation does impose the limit of only evaluating nesting patterns within the principal orientations of the grid.

Although some of the systems described above perform reasonably quickly, the processing times would be considerably higher if they were adapted to deal with a large, varied parts list which had to be nested onto a stock sheet. In addition a suitable method of sequencing the parts would have to be devised. These algorithms are efficient in their limited domain, however their methods could not be successfully applied to the 'low volume, high variety' production environment, which is the area of study for this research.

2.6 Irregular Part Tiling (Paving)

The Paving or Tiling of shapes is often referred to as tessellation and consists of placing shapes in a repeated pattern, with no constraint on the orientation of any individual shape, such that there are no gaps between the shapes. There are only a small number of convex straight line shapes which can form a tessellating pattern on their own.

Kershner (1968) defined three types of hexagonal pavers and eight types of pentagonal pavers which will tessellate perfectly. These, in addition to any type of triangle or rectangle, are the only convex straight sided pavers which will tessellate. By including concave straight line shapes and shapes which include curved sections the number of potential pavers which will tessellate is huge. Grunbaum and Shephard (1977) have expanded Kershner's list of pavers by including concave and curved shapes, however this list is by no means exhaustive. There are also a vast number of tessellating patterns which can be formed by using two or more shapes in combination. By enclosing a part within a shape which can form a tessellating pattern it is possible to create a repeated pattern of parts on a sheet.

Dori and Ben-Bassat (1984) describe their system for placing a number of identical parts on a sheet by paving. The system first takes the convex hull of the part which is then enclosed within a hexagonal paver which can tessellate. It is also possible to enclose a pair of parts within the paver. The paver is then repeated across the sheet to either fill it or to give the required number of parts. If a pair of parts are not to be enclosed it may be quicker to enclose the part within the paver directly rather than going through the conversion of the part to its convex hull. An enclosure efficiency of 93% is given, however the overall efficiency of this system will be much lower due to the perimeter waste incurred at the interface between the edges of the tessellating patterns and the sides of the sheet.

Koroupi and Loftus (1991) have furthered the work carried out by Dori and Ben-Bassat by creating a system which can create a paver for any part consisting of lines

and arcs. This system also takes the convex hull of the part, but the arcs are approximated by chords of a user defined length. The side numbers are then reduced to reach a six sided polygon. The polygon may need to be further altered to conform to a shape which will tessellate. This paver is then repeated across the sheet in the same way as the system by Dori and Ben-Bassat. Both of the nesting systems described are limited by only attempting to use one generic paver shape from an almost infinite range. If a large number of the same part were to be cut from sheets (rather than the more common stamping from strip) it might be economically viable to search a vast range of tessellating shapes manually or go through a process of trial and error with the unsimplified part shape. However, for lower volume production, a more limited range of tessellating shapes could be searched for the shape most suited to the part to be nested. This would be the most appropriate expansion of the systems described.

2.7 Sheet Size Optimisation

The systems described so far aim to optimise the placing of parts onto stock sheets, often for a specifically bounded situation. The work of Qu and Sanders (1989) introduces the additional factor of optimising the sizes of sheets used, where more than one stock size is available, regardless of the method of nesting the sheets. This is of particular value where the parts are large relative to the stock sheets, which in turn limits part combinations and, in general, generates greater waste. This paper suggests three algorithms to optimise the combination of stock sheets used. The first, a branch and bound method, operates by retaining the 'best so far' nodes while exhaustively searching all combinations. The second method takes an arbitrary first path through the nodes and then exhaustively backtracks through all other options. Both these methods supply an optimal solution at the expense of considerable computation time. The final method is a 'greedy', non-backtracking algorithm which discards all but the optimum node at each stage. This is considerably faster than the previous algorithms, with the drawback of not guaranteeing an optimal solution. This work is particularly useful as the methods developed can be applied to any nesting system, providing more than one stock sheet size is available. The system could also be used to optimise the use of off-cuts from previous production runs.

Linear Programming (LP) systems have been developed to establish an optimum sheet size, or range of sheet sizes, from which customer specified sheet sizes can be cut with minimum waste [Chambers and Dyson (1976), Beasley (1985) and Tokuyama and Ueno (1985)]. These are intended for use by sheet rolling mills and sheet stockists and cannot be applied to improve 2-D nesting layouts.

2.8 Three Dimensional Packing Systems

The 3-D packing systems encountered are aimed at either packing shipping containers or loading pallets and all systems are constrained to the placing of rectangular boxes only. Purely theoretical work has been carried out to examine possible exact divisions or combinations of cubic areas. Gilmore and Gomery (1965) adapted a 1-D LP method to determine the optimum way to cut a number of small rectangular sections from a larger piece of rectangular stock material. This method is highly constrained and would not be suitable for planning container or pallet layouts. Gardner (1979) describes work carried out to prove the number of smaller cubes of certain sizes that a large cube can be divided into. Such mathematical proofs are of little practical value for packing as they only consider waste free divisions and do not suggest any rules or methods to aid packing.

George and Robinson (1980) first formulated an algorithm to tackle the packing of a container with up to 20 different types boxes. A number of criteria are used to determine the placement order of the boxes. The boxes to be packed cannot be constrained in orientation or position. Boxes are packed in vertical columns from the back left corner of the container to form a wall and this process is repeated until all the boxes are positioned. Gehring et al (1990) also use a wall building strategy, however the boxes are priority ordered by volume and the wall is constructed in layers. Bischoff and Marriott (1990) furthered the work of George and Robinson by adapting their system to use 14 different heuristic rules to sequence box placement. The objective in each case was to minimise the length of container required to accommodate a given amount of cargo.

Dowsland (1991) has carried out a survey of container packing systems and examines the possible strategies which could be used to design algorithms or improve them. The systems which were examined either tried to emulate the way containers are manually loaded with no planning or carried out wall building. Wall building effectively converts the problem into a series of 2-D layout problems.

Ngoi et al (1994) have developed a method of container packing which departs from the wall approach described earlier. The first box is placed at the back corner of the container and the area around it is divided into rectangular volumes. As each subsequent part is placed all the available areas in the container are checked and the smallest to accommodate the part is taken. This approach avoids free areas becoming blocked by parts. The vertical dimension of each box is fixed but it can rotate in the X and Y dimensions. All boxes are free to be placed in any position and all the boxes are assumed to have the same customer destination.

Hodgson (1982) developed the IPLS (Interactive Pallet Loading System). This system uses a linear programming algorithm to generate a 3-D pallet plan consisting of either layers of boxes of a similar height or columns of boxes of a similar length and width. Account can be taken of the weight of boxes and groups of boxes for the same destination can be kept together. The generated plan can be further improved by the systems interactive facility.

Many approaches in the 3-D environment are taken from systems and methods developed for 2-D layout problems. Due to the additional complexity in the container packing and pallet loading problems the systems to solve them require considerable constraints to be imposed. Thus it is unlikely that methods for the 3-D packing problem would prove better in the 2-D environment than the existing methods.

2.9 Other Layout Problems.

Part of the design of microchips is concerned with the positioning of rectangular chip elements into the rectangular area of the chip. No automated system currently exists to plan chip layouts. This is due to the complexity of planning wire routes between elements, the necessity to cluster elements together, the vast number of elements present in chip designs (10000+) and the precedence of a logical clear layout over a tightly packed one. Chip layouts are thus created directly by the designers using interactive planning systems. Bentley et al (1980) describes a system to automatically check layouts for faults. The layout is scanned from 'top to bottom' to find intersections between chip elements and check that they are intentional. It also checks that a sufficient gap has been left between each element. This layout problem differs greatly from the sheet metal parts layout problem and there does not appear to be any methods which can be transferred across.

The Facilities Layout Problem is concerned with the spatial location of the functions which combine to form a production or services facility. The elements in a production facility might be production cells, a tool store, a material store, an assembly area and production control offices. Material and information will flow between any pair of functions to different degrees, the optimal layout is one where this flow is minimised. Traditionally this problem has been tackled in one of two different ways: either quantatively, by minimising the flow cost, or qualatively by maximising the closeness rating. In the Facilities Layout Problem the layout size is often pre-set by the size of an existing building or if the overall layout size is to be constrained it is only one of a number of constraints to be taken into account when meeting objectives.

Kusiak and Heragu (1987) have provided a comprehensive review of the existing procedures to solve the Facilities Layout Problem for a single objective. The optimal procedures include those based on a 'branch and bound' method [Lawler (1963)] and the 'cutting plane algorithm' [Burkard and Bonninger (1983)]. Both these methods are reported to require a large amount of computation time and are thus impractical for large problems. A quicker heuristic method is proposed by Golany and Rosenblatt

(1989), which constructs a 'first attempt' layout and then attempts to improve it by swapping functions. This method does not guarantee optimality. Malmborg (1994) has developed an interactive analytical layout system with a heuristic to provide initial layouts which can then be improved by the user. Raoot and Rakshit (1994) have developed a 'fuzzy heuristic' to tackle the multiple objective nature of the Facilities Layout Problem, however all facilities are assumed to be of the same size and perfectly mobile. Sirinaovakul and Thajchayapong (1994) have developed a facilities layout system which generates a number of good solutions using a closeness heuristic and a pattern allocation system. Throughout the process an expert system is consulted to select appropriate proximity weightings for pairs of functions. Negative values can be entered to keep functions apart. Schwarz et al (1994) have developed a system, based on a graph-theoretical model, to automatically generate architectural layout plans of new buildings. The system is reported to run quickly, however it has only been applied to relatively simple problems.

In the cutting stock problem the location of one part in relation to the other parts is not relevant and minimising the area which the total layout occupies is the sole objective. The radical difference in the constraints imposed on these two types of layout problem result in little transfer of methods being possible between the two fields.

Chapter 3

Design of a New Nesting System

In this chapter the design of a new system to nest sheet metal parts, based on the method of enclosing parts within rectangular envelopes prior to placement, is described in detail. However, the actual system developed in this work, described in Chapter 4, is limited to demonstrating and evaluating the rectangular envelope placement algorithms (Sections 3.6, 3.7 and 3.8).

3.1 Requirements of the System

The planning costs of high volume components are very low per unit, and the material costs relatively high. Experienced human planners are very effective at creating efficient layouts in this environment as the time and cost of 'trial and error' planning is economically acceptable. It is also true that an automated planning system in this environment could also use time consuming exhaustive procedures to generate good layouts. The economics of 'high variety, low volume' manufacture limit the resources which can be applied to the planning of individual products, as planning is a major proportion of unit cost. Manual planning is not ideally suited to this production environment. Minor layout inefficiencies, due to simplifications of the problem or the parts, are acceptable if they reduce the planning cost or lead time. As the needs of the above two production volumes conflict, the system must be developed specifically for one of these two areas. The 'high variety, low volume' manufacturing environment would seem to benefit more from an automated nesting system and it is to this area that the new system is aimed. Thus the new system designed is required to nest parts lists with a large variety of parts quickly and with a low level of waste.

Many existing nesting systems attempt to position the largest parts in the layout as early as possible. This is because the spaces remaining on a semi-complete sheet may be too small to accommodate large parts. These spaces would have to be accepted as waste and a new sheet started to accommodate the remaining parts. If the larger parts are positioned early in the nesting process, there are still plenty of smaller parts to fill the gaps which remain around them. Thus it is a good policy to give a high priority to the placement of large or long parts and this policy is adopted for the new system designed. For similar reasons each part should be placed at the perimeter of the remaining sheet area to retain the largest possible free area. Existing systems generally commence part layout creation from the bottom left corner of the sheet. As there is no apparent disadvantage incurred with this start point over any other corner, the new placement algorithms designed also start part placement in the bottom left corner of the sheet.

Almost all cutting processes require a 'bridge gap' to be placed between each part on the sheet. This is either to account for material removed by the cutting process or to preserve local sheet stability during cutting. This is critical to the generation of a good layout for the chosen cutting process. The new nesting system places a specified 'bridge gap' between each part in the layout. The 'bridge gap' can also be placed around the sheet perimeter if the sheet edge finish is not acceptable for a parts edge.

Allowing parts to be rotated increases the number of possible layout solutions and thus improves the potential layout efficiency. The orientation of most parts relative to the sheet grain is of little importance as most sheets have little difference in strength due to orientation. However some specific formed features or bends may require the part to have a specific orientation on the sheet (this is an important facility when planning layouts for the cutting of patterned cloth in the garment industry). A number of existing systems have this facility such as CASNS [Prasad and Somasundaram (1991)]. Nee et al (1984a) constrain bend axes to exceed a 45° angle to the sheet grain for strength purposes. Thus the new nesting system designed includes the facility to hold selected parts in a specified orientation.

A layout which does not conform to the 'guillotine cut constraint' will invariably be more efficient than one which does and the relaxation of this should be utilised whenever possible. However, if a guillotine shear is the only cutting process available the layout produced must conform to the inherent constraints of this process. Thus the new system is also capable of generating 'guillotine cut constrained' layouts.

Due to roller deflection during sheet manufacture there will be very slight differences in thickness across the sheet's width. Generally sheets are slightly thinner along their edges than they are in the middle, although this may be reversed if the roller deflection has been over compensated for in the tooling. These differences are very small with modern sheet manufacturing methods so a component's thickness tolerance would have to be exceptionally tight for a problem to occur. No existing system encountered has a mechanism for placing parts with consideration for differences of thickness in the sheet. This does not seem to be a sufficient problem to justify its accommodation in the new nesting system designed.

Some nesting systems allow flawed areas on the sheet to be marked and avoided during part nesting. Nee et al (1986) developed a system with this feature which also allowed irregular sheets to be nested. The marking of the flawed parts of a sheet is particularly important when cutting certain types of cloth. The accommodation of irregular sheets is critical for cutting leather parts from hides. Although these features are far less important in the sheet metal environment, they would be useful when dealing with damaged sheets or irregular 'off cuts'. Therefore the new system designed includes a method of nesting irregular sheets.

It is desirable for the new system to be easy to up date and improve, thus a modular format is used wherever possible. This will help to keep the effects of any future changes localised. A modular format also allows unwanted functions of the total system to be removed. For instance, a company may feel that for their product range a particular function is better carried out by a human planner. The input data required must be as simple as possible to allow easy interface with CAD, as must the output generated, for interface with graphics and CAM packages.

3.2 Approach of the System

The complexity of most sheet metal parts would require a large amount of data manipulation to allow parts to be nested in their detailed form. Thus some level of simplification is required, resulting in a play off between the level of part simplification and the ease of part manipulation. Thus the new system designed adopts a two stage approach, with all the parts being initially simplified and then these simplified representations being manipulated to create the sheet layout.

A straight line envelope (Figure 1.16, Section 1.8) is the most complex representation for which the automated selection (with a short processing time) of the next part to add to the layout is considered to be possible. However, this level of simplification still requires a very sophisticated set of rules to determine the next part to place and to recognise its optimum position and orientation. If the part is represented by a rectangular envelope, only two side length dimensions need to be stored and only two possible principal orientations need to be considered. Therefore in the new system designed all the parts are simplified to their smallest enclosing rectangular envelopes. With the bridge gap added to the rectangular envelope's dimensions, this format allows simplified part location as the envelopes can always be placed in contact.

As enveloping all the parts individually may not prove particularly efficient, more than one part can be placed in each envelope. This requires parts which can be positioned together in an efficient pattern to be identified and grouped. A small group of identical parts may be 'paved' or 'tiled' to form an efficient tessellating pattern which can subsequently be represented as a rectangular envelope. Alternatively, any non-identical group of parts in the list may be found to form a cluster which fits within a rectangle efficiently. Either form of grouping could be carried out manually by the operator, using an interactive system, or an automated system could be developed to carry out this task. The enveloping of individual parts or groups of parts could also be carried out either manually or automatically. These elements of the new system are described in detail later in this chapter.

Many of the existing algorithms which have been applied with success to the problem of placing rectangular envelopes or parts onto rectangular sheets use a common strategy. This is to take each part in a pre-determined sequence and add it to the existing layout according to the individual rules employed. Any pre-determined sequence (eg. decreasing height order) can prove to be far from optimal for individual problems and only assures that worst case performance does not occur. There does not seem to be any obvious improvement of significance which can be made to the part placement rules employed and the only remaining way of improving nesting performance is to exhaustively try every part placement sequence. However, it has been established in Section 1.4 that any exhaustive search of part placement sequences is impractical for the nesting of large lists of parts. Therefore, the new nesting algorithms try to fit either individual or sets of parts to the sheet according to matches in their dimensions, rather than using a pre-determined part placement sequence.

Two new part placement algorithms and a layout improvement algorithm have been designed and are employed in the new nesting system [Scott and Mileham (1993a), (1994a) and (1994b)]. It is considered that all of the new algorithms adopt a novel approach to the nesting problem. At each stage of each placement algorithm's operation the parts list is searched for the most suitable part or group of parts to place. The **Strip Fit** algorithm aims to create efficiently nested strips of parts which span the sheet. The **Area Fit** algorithm selects the most suitable individual part to be located in the next free area of the sheet. Both of these algorithms create layouts which conform to the 'guillotine cut constraint'. The **Strip Squeeze** algorithm improves the layout created by the Strip Fit algorithm by removing its adherence to the 'guillotine cut constraint'. The Strip Squeeze algorithm can also be used to improve any layout which consists of strips of parts which span the sheet. The Strip Fit algorithm and the Area Fitting algorithm can either be used individually or sequentially to produce a layout.

Thus a complete new nesting system is proposed. The details of the parts are imported from a CAD system. The parts list is interrogated for any parts which will form efficient clusters or efficient tessellating patterns (tiling or paving) and these patterns

are created. These patterns, along with the remaining individual parts, are placed within rectangular envelopes and these envelopes are then passed to the layout generating part of the system. This uses the Strip Fit algorithm, the Area Fitting algorithm and the Strip Squeeze algorithm to generate complete sheet layouts. The complete system is shown in Figure 3.1.

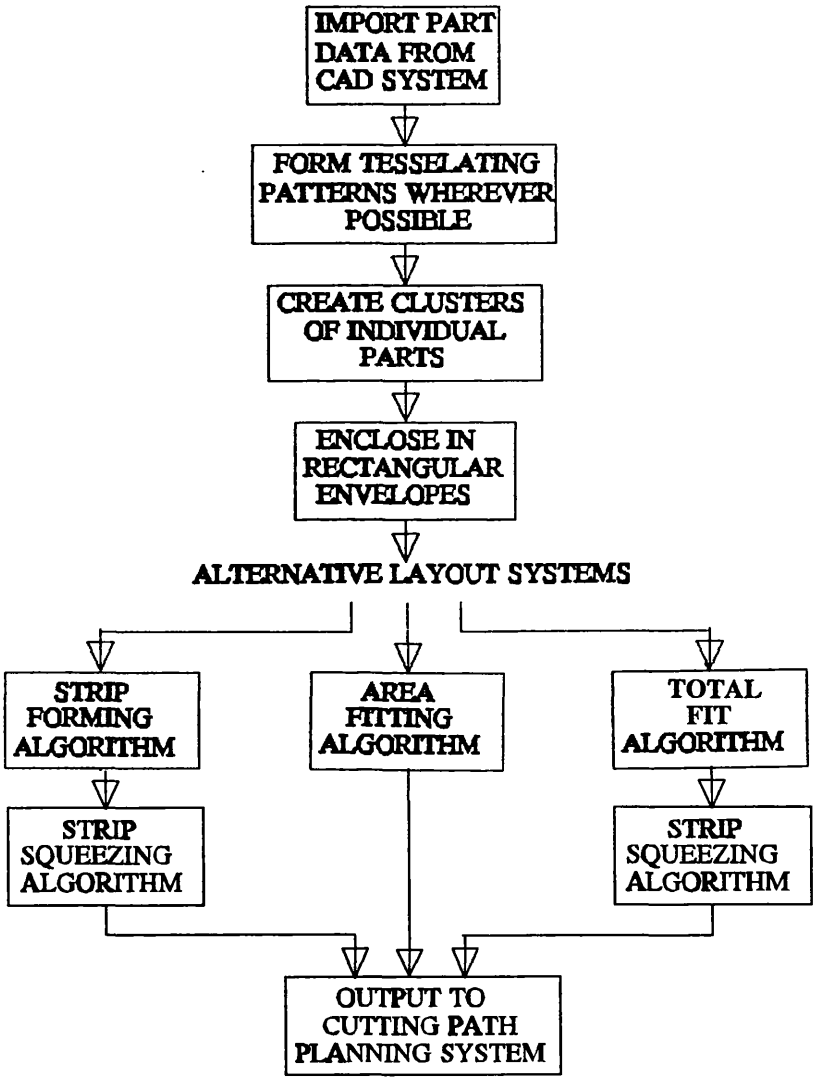


Figure 3.1.
Flow Diagram of the Total Nesting System.

In summary, the features of the new system designed are:-

- To create efficient clusters or tessellating patterns of parts where possible.
- To place these patterns and clusters, and any individual remaining parts, within the minimum enclosing rectangular envelope.
- To create a nested layout of these envelopes on the specified size of sheet.
- To produce 'good' rather than optimal solutions, to keep the required processing time low.
- To be capable of nesting a large number of non-identical parts.
- To add a specified bridge gap around each part to be nested.
- To hold parts in their given orientations or to permit the parts to be placed in their other primary orientation.
- To create layouts which conform to the guillotine cut constraint, if required.
- To be capable of creating layouts on irregular sheets.
- To have a modular format which allows discrete areas of the system to be replaced or modified with no effect on the operation of the other areas.

The critical area of any nesting system is the part placement algorithm. Three new nesting algorithms, which have been developed, coded and tested, are described in detail in Sections 3.6, 3.7 and 3.8. The various peripheral areas which are required to create a full commercial system have been conceptualised and are described in Sections 3.3, 3.4, 3.5 and 3.9. The detailed design and coding of these later areas of the new system are considered as further work.

3.3 Enveloping Parts within Rectangles

The parts to be nested may be entered into the nesting system from a CAD system or from a solid modeller. In either case the dimensions of the minimum enclosing rectangle for each part can easily be found. If the component drawing shows bends, an unfolding algorithm will also be required to generate the equivalent 'flat' shape. The drawing is searched for the highest and lowest points within it and the points furthest to the left and to the right. By finding the difference between these pairs of points the X dimension and Y dimension ranges of the part can be established. These are the dimensions of the smallest rectangular envelope which will completely enclose the part. However, ensuring that this is done with the part in its optimum orientation is more difficult. This could be overcome using a trial and error method with a number of orientations, as described by Freeman and Shapira (1975). Alternatively, a heuristic method could be used, such as aligning the envelope with the part's longest side or the parts could also be entered into the system in a pre-determined orientation. A planning engineer, using an interactive system, would be able to place each part in a good or optimum orientation in a short time.

The bridge gap around a part is either the width of the material removed by the cutting process or a space left to give the sheet stability during cutting. If the perimeters of all the parts are expanded by the entire bridge gap required, the parts will be separated by two gaps, unless the system can account for this and allow the gaps to overlap. It is considered that it is preferable to expand each part by half the required bridge gap so that contact between the expanded part dimensions ensures the correct bridge gap. This alone would then cause a problem at the sheet perimeter, where half a bridge gap would separate the parts from the edge. Moving the representation of each sheet edge 'out' by half a bridge gap would result in the actual part lying flush with the sheet edge when placed by this method. However, in some cases the edge of the stock sheet will not have a suitable finish requiring the part to be separated from the sheet edge by a full bridge gap. This can be achieved by moving the representation of each sheet edge 'in' by a half bridge gap.

3.4 Part Clustering System

Many parts have shapes which result in a high levels of waste when they are enclosed in a rectangular envelope. If such parts are placed in clusters with one or more other parts, a combined form can be created which gives less waste when enclosed in a rectangular envelope. A number of good systems exist to do this which have been described in Section 2.4. The systems by Parry-Barwick (1995) and Adamowicz and Albano (1976a) are particularly appropriate. However these systems require a considerable amount of processing time to create good part clusters which may be unacceptable for some applications. This process will also create a number of larger envelopes which are generally harder to nest.

Thus a compromised clustering system may be useful which sacrifices some material efficiency to increase the planning speed and prevent particularly large clusters of parts from being created. One approach would be to identify parts which have large free areas within their rectangular envelopes into which other parts could be placed without violating the bounds of the envelope. This would avoid any increase in the size of the envelopes to be subsequently nested. The simplest method would be to identify free rectangular areas within envelopes. This removes complex part manipulation as the largest part which will fit into the space, with account being taken of the required bridge gap, will be selected. This is easily done as all the parts to be nested are already enclosed within rectangular envelopes.

If the part is represented as a solid model it is easy to recognise areas in which a prospective part could be placed. However solid modelling is more expensive, complicated and time consuming than traditional 2-D CAD drawing. There are two simple methods which could be used to identify prospective areas for a part to be positioned in. From positions around the perimeter, parametric rectangles could be expanded towards the part, halting when they hit the perimeter of the part, see Figure 3.2. This has the draw back of not identifying large holes in parts into which another part could be nested. A system which could identify large internal features would operate by placing a grid over the part. By identifying the intersections of these grid

lines with lines portraying the part, the areas of solid part and of free space could be found, see Figure 3.3. Finally, human planners excel at this type of nesting, where a limited number of parts are to be manipulated, so an interactive system may be preferred.

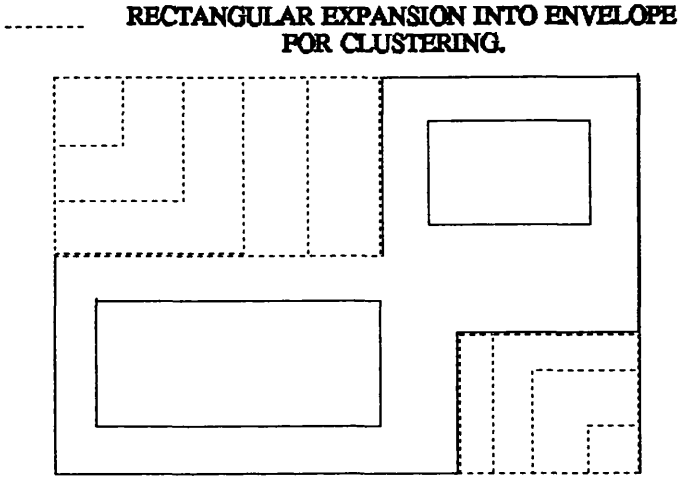


Figure 3.2.
Expanding Rectangle method of identifying free areas for part placement.

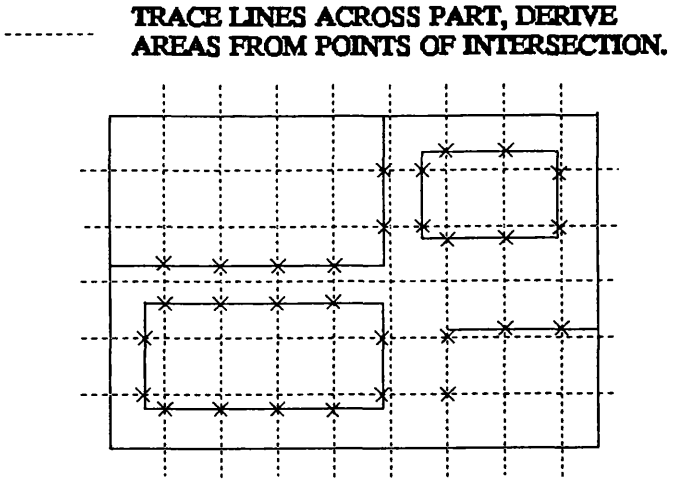


Figure 3.3.
Trace Line method of identifying free areas for part placement.

3.5 Part Paving (Tiling) System

The most efficient layout for a large number of identical parts is generally a repeating symmetrical pattern, known as a tessellating layout. This type of layout is normally created by a planning engineer recognising an efficient orientation and position for a small number of the parts. From this small group a tessellating pattern emerges which can then be used to dictate the placing of the remaining parts. The planner may have developed a set of informal rules over the years to do this, or may simply take a trial and error approach. Unfortunately, this type of spatial reasoning is very difficult to carry out automatically and invariably the problem must be simplified to some extent.

Often the repeated pattern of a part nested this way takes the form of rows and columns. In this situation the sheet can be divided into uniform rectangles; each enclosing a part and an identical section of the inter-part waste. The layouts created with a square paver enclosing a circular part in Figure 3.4 are of this type. For some sheet sizes the most efficient pattern to nest circular parts is to use what appears to be off-set rows. In such a layout the parts of each subsequent row are vertically aligned between the parts of the previous row. In fact this pattern can also be divided into equally sized pavers, in this case of a hexagonal form, also shown in Figure 3.4. Although some tessellating shapes may be very complicated, there are many simple shapes which will form tessellating patterns, such as 'L' shapes, diamonds and chevrons.

Thus a set of identical parts can be nested on a sheet with little waste by finding a suitable enclosing paver which forms a tessellating pattern. Unfortunately, the best paver to use cannot be chosen simply on the basis of the waste around the part within the paver. This is due to the additional waste which may be encountered due to a mismatch between the tessellating pattern and the sheet edges. This is shown in Figure 3.4. The hexagonal paver pattern gives the best layout for sheet 1 and the square paver gives the best layout for sheet 2. Thus the optimal paver will depend on the sheet size as well as the geometry of the part.

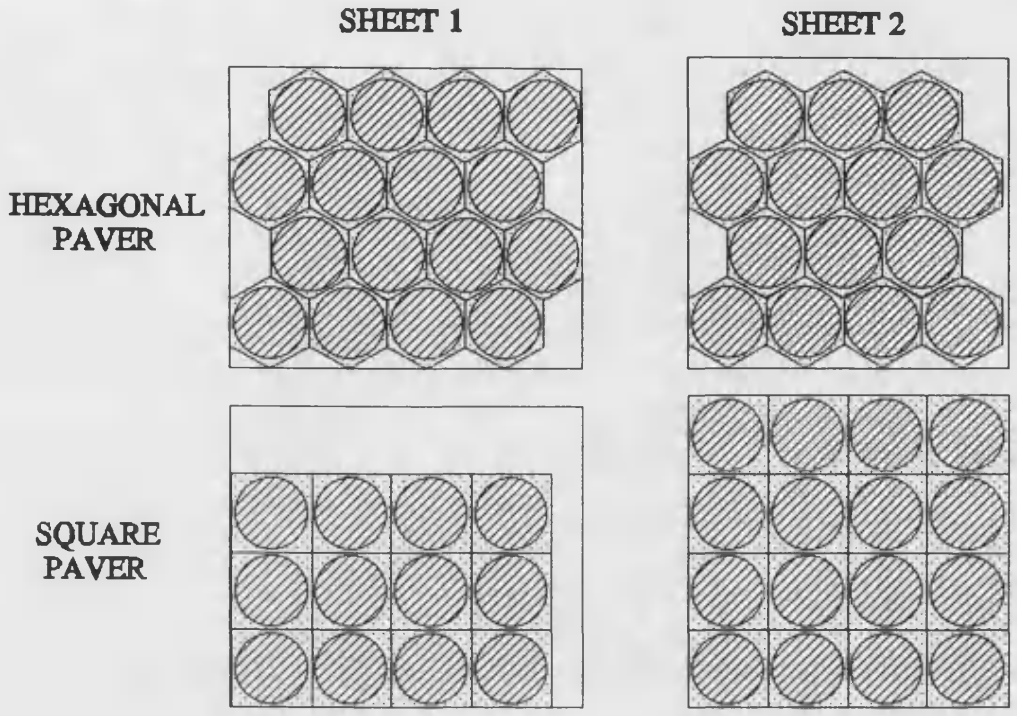


Figure 3.4.
Square and Hexagonal Paved Layouts for discs on two sheet sizes.

If the tessellating pattern is to be enclosed within a rectangular envelope for subsequent nesting, only the sheet's dimensions need be considered to ensure that the dimensions of the envelope around the tessellating pattern do not exceed the dimensions of the sheet. If a rectangular envelope is to be placed around a tessellating pattern there will usually be some perimeter waste, in addition to the waste in each paver, which must be accounted for when selecting the optimum paver. An example of this is to consider the edges of Sheet 1 (Figure 3.4) as the minimum enclosing rectangular envelope bounds of the hexagonal paver layout. An example of a situation in which the minimum enclosing rectangular envelope will not include any perimeter waste is to consider the edges of Sheet 2 (Figure 3.4) as the minimum enclosing rectangular envelope bounds of the square paver layout. Even with a rectangular paver there may be some perimeter waste if the number of parts cannot form a 'perfect' pattern (ie. 3 by 3 or 5 by 4). Perimeter waste is not normally directly proportional to the number of parts. The pattern needs to be created and enclosed by the minimum outer rectangle to allow perimeter waste to be accurately calculated.

Figure 3.5 shows the perimeter waste associated with four different tessellating layout sizes created using the same hexagonal paver. The perimeter waste per part decreases as the number of parts increase, making the larger layouts more efficient than the smaller layouts. In this example the envelope aspect ratio is constant. However, perimeter waste will also increases if the aspect ratio of the enclosing envelope is increased, for the same envelope area. This is due to the relationship between perimeter and area. For instance, a 2 by 2 envelope has a perimeter of 8 units, but a 4 by 1 envelope, has a perimeter of 10 units.

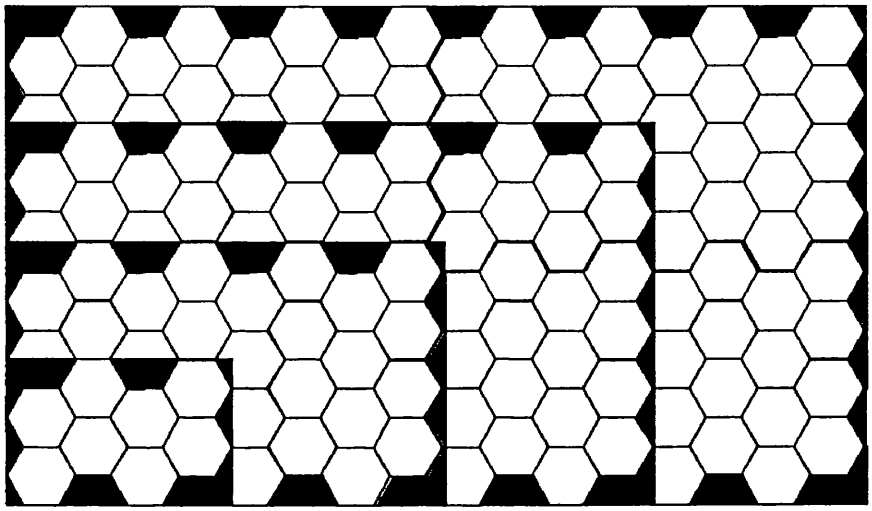


Figure 3.5.
The perimeter waste incurred with different sized layouts.

The existing paving systems were reviewed in Section 2.6. These are limited to placing parts within hexagonal pavers of varying dimensions. Most of the rectangular envelope nesting systems discussed in section 2.2 would produce a tessellating pattern if given a number of identical parts to place. However, there are shapes other than hexagons and rectangles which would make good pavers for use in a nesting system. Four more pavers are introduced in Figure 3.6 along with the tessellating pattern they form and their associated perimeter waste. In addition, Figure 3.7 shows a further four shapes which can be paired to create either a rectangle or parallelogram, for which there is an established tessellating pattern. It must be noted that some of these patterns require the part to be rotated and therefore may not be applicable to parts with a fixed orientation relative to the sheet.

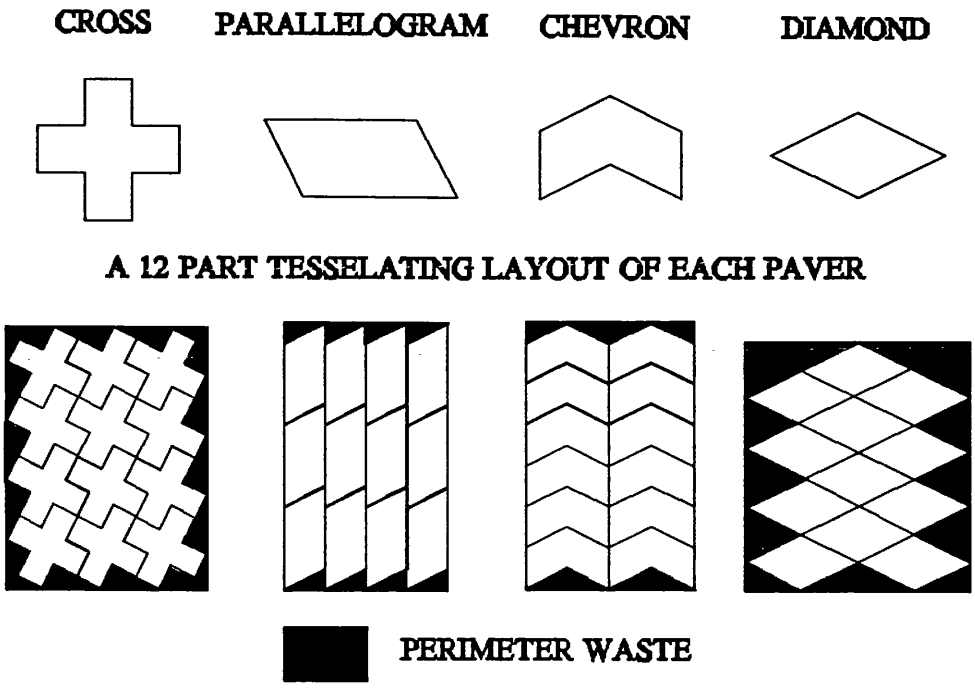


Figure 3.6.
Other simple shapes which form tessellating patterns.

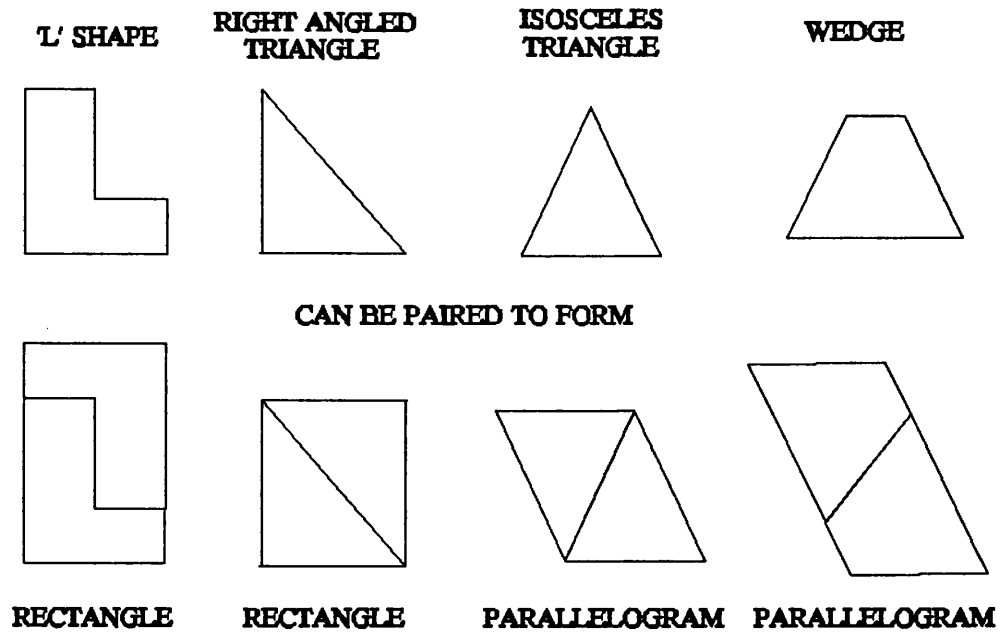


Figure 3.7.
Some other shapes which can be paired to form an existing paver shape.

An automated nesting system could use a small number of paver shapes to give a choice of tessellating patterns. This would often lead to better layouts containing less waste. All the pavers would be held as parametric shapes which would be constricted around the part to be nested. Thus the smallest possible paver dimensions to enclose the part would be found. At this stage the waste within some pavers may be so high that it is inconceivable that they will give the optimum layout. These pavers can be discarded. The most efficient pattern for each remaining paver is found that accommodates the number of parts to be nested. If the dimensions of the patterns produced are considerably less than those of the sheet to be used, the pattern with the least waste is chosen. If the pattern exceeds the sheet dimensions or lies close to a sheet side, producing a strip too narrow for further use, another method of layout evaluation must be used. This is to overlay the tessellating patterns onto the sheet in order to find the one which will accommodate the most parts. This is because slight differences in sheet dimensions can alter the best pattern to use (Figure 3.4).

If the new system is to avoid evaluating each prospective layout, a method of forecasting their perimeter waste must be found. One method would be to maintain a database with the layout dimensions of all possible paver multiples (within an upper limit) and the positions of the parts in them. Alternatively, a calculative system could be used. This would contain equations, in terms of the paver's dimensions, which would give the layout dimensions for the number of parts entered.

Figure 3.8 shows an hexagonal paver with its critical dimensions, denoted h, w and s. Figure 3.9 shows an example layout and a representation of the same layout with parts enclosed in rectangles. This allows easier comprehension of the way the layout dimensions expand. If such a layout has a number of columns N_{col}, the maximum number of parts in an even numbered column is N_{even} and in an odd numbered column is N_{odd}. The total layout width X and the height Y can be calculated by the following equations:-

$$X = w + (N_{col}-1)(s+(w-s)/2).$$

If $N_{odd} > N_{even}$ $\quad Y = N_{odd}h.$

Else $\quad Y = N_{even}h + h/2.$

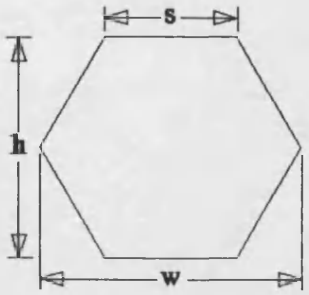


Figure 3.8.
The critical dimensions of a hexagonal paver.

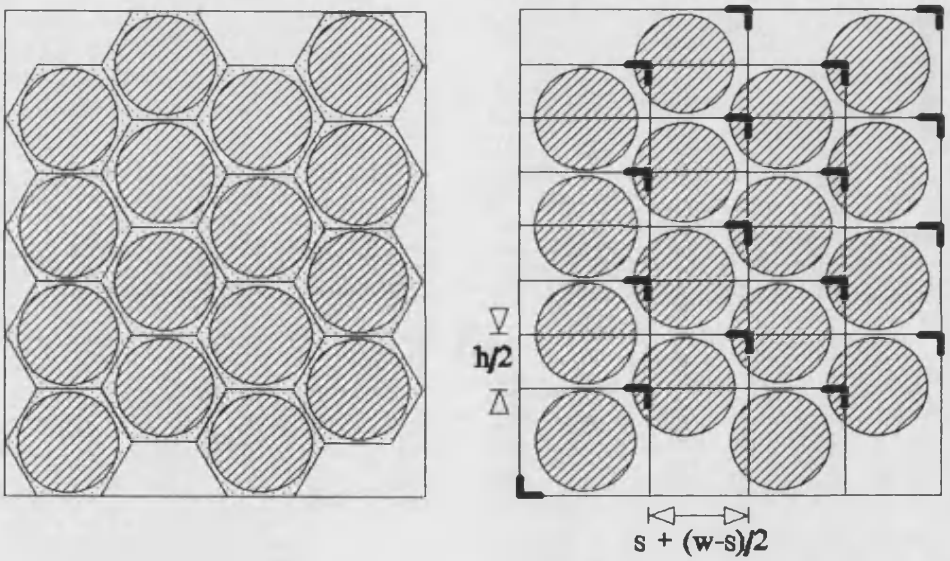


Figure 3.9.
Method of calculating the size of a tessellating pattern.

There are reasons other than reducing material waste levels to include a sub-system to produce tessellating layouts within an overall nesting system. If groups of identical parts are not paved they will be given individual rectangular envelopes in the main system, which will increases the processing time. In addition, the main system may place the parts on a number of sheets. Keeping identical parts clustered together reduces tool changes, tool travel and allows easier grouping of parts into batches for subsequent processing.

3.6. Strip Fit Algorithm

This algorithm operates by selecting a combination of part envelopes from the parts list which can be positioned on the sheet to form a neat strip. To keep waste low the envelopes must be of a similar width and the sum of their lengths must be close to that of the sheet dimension across which the strip is to be placed. This algorithm only deals with parts which have been simplified to rectangles or enclosed within a rectangle. When parts are referred to in this section they are always in their simplified rectangular envelope form.

Consider the situation of creating a strip along the Y-axis of an empty sheet. None of the parts in the list can be rotated and no part has an X dimension greater than the sheet's X dimension. The first step is to identify in the parts list, a sub-group of parts which have similar X dimensions. This reduces the problem of creating the strip to one of finding a good combination of parts to make a suitable length strip ie. the problem has been reduced to one dimension. Thus the chosen parts Y dimensions must sum to a value less than, but close to, that of the sheet's Y dimension.

The parts list is sorted into decreasing (technically non-increasing) order of their X dimensions. An X dimension range must be set to determine which parts will be accepted into the sub-group. Initially, the upper bound of this range is the largest X dimension in the parts list and lower bound is the upper bound adjusted by an acceptance tolerance. If the largest dimension were 500mm and the acceptance tolerance were 10%, any part with an X dimension between 500mm and 450mm would be accepted into the sub-group. The Y dimensions of the sub-group members would then be summed to check that sufficient parts were present to make a complete strip. If there are insufficient parts or a strip of suitable length cannot be made the next largest X dimension in the list is taken as the range upper bound and the original largest X dimension is put to one side. This process is repeated until all X dimensions in the parts list have been tried or all the parts in the list are placed.

Given that sufficient parts have been placed in the sub-group to form a strip of acceptable length, the remaining task is to find a suitable combination of parts which closely match the required strip length. The best strip length fit could be found by an exhaustive search of all part combinations. However, this approach would not prioritise the large parts in the sub-group and exhaustive searching is also time consuming for large sub-groups. A better approach is to use an acceptance tolerance for the required strip length to determine when a suitable combination has been found. The upper bound of this tolerance is the sheet Y dimension and the lower bound is set at a pre-determined percentage below this. If a combination of parts is found with lengths summing within the range, then they are removed from the sub-group and nested. Alternatively, a number of strip combinations could be evaluated, the best retained and this combination used if its length fell within the acceptance range.

Figure 3.10 shows the former of these two methods for creating a strip of parts. All the parts with an X dimension within the strip width tolerance have been removed from the main parts list and placed in a sub group in order of decreasing height (working from the bottom up in the diagram).

(I) Shows the length and width tolerances for the strip. These have been derived from the sheet's Y dimension and the X dimension of the largest part in the sub-group.

(II) and (III) Parts A and B are placed in the strip and, in each case, it is checked that they do not exceed or meet the strip length tolerance.

(IV),(V) and (VI) Part C, D and E are tried, but C and D exceed the strip length tolerance and E causes the strip to fall short of the tolerance.

(VII) To allow further envelope combinations to be tried, part B is removed and replaced by part C. There is no need to check the strip length tolerance as the Y dimension order of the sub group assures that B is bigger than C.

(VIII) Part D is added and a combination which meets the strip length tolerances is found. If there are other combinations which meet the tolerance they would not include as many large parts, however they may provide a more efficient layout. However, this first strip is accepted to avoid the need for an exhaustive search of all

strip combinations and to increase the efficiency towards the end of the nesting by placing large awkward components first.

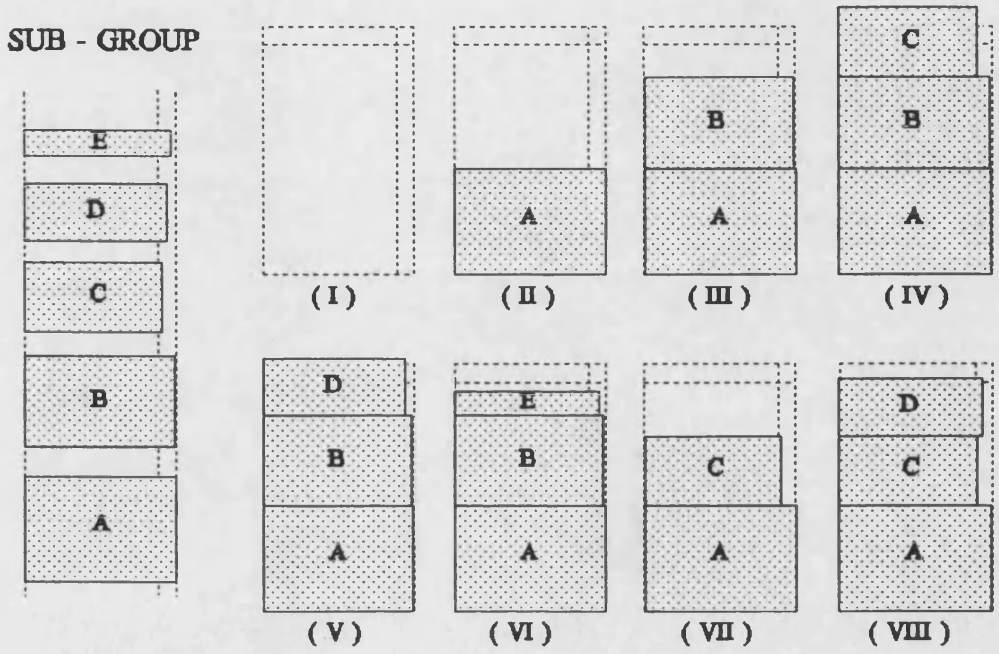


Figure 3.10.
The step by step creation of a strip of parts.

Figure 3.11 shows a completed layout created by the 'Strip Fit' algorithm. The broken lines represent the acceptance tolerances for the length and width of the strips. The system tries to form wide strips first in order to place the larger parts as soon as possible. Within each strip the parts with the largest Y dimensions are placed first in the strip. This guarantees the largest parts have priority as all the X dimensions are approximately the same due to the strip width acceptance limits. The system then checks the remaining parts in the sub-group for other satisfactory combinations. If none are found those remaining are returned to the parts list, a new strip width range is set based on the next largest X dimension and parts with a dimension within this range are searched for.

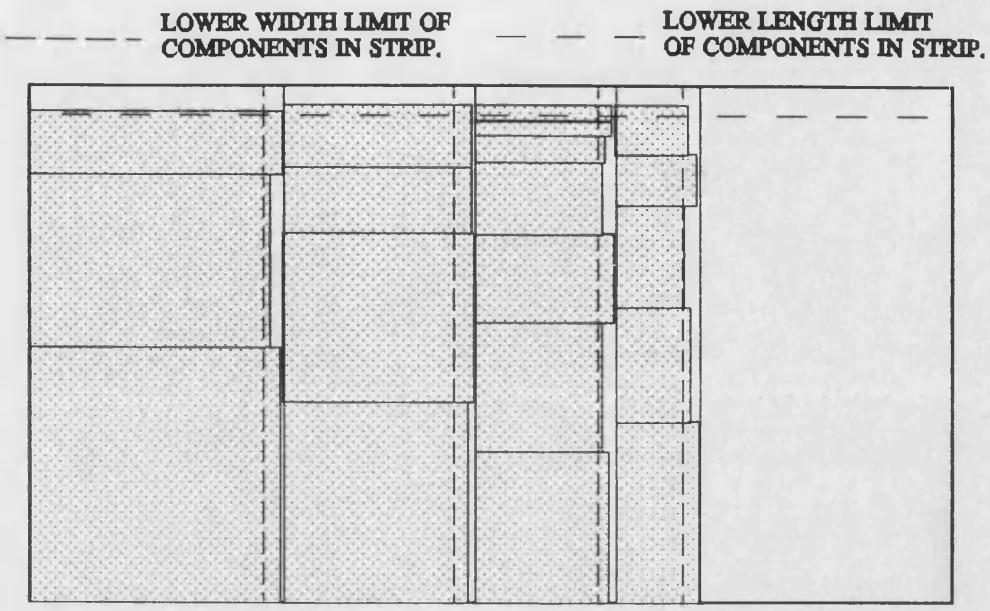


Figure 3.11.
A layout created by the Strip Fit algorithm.

If a strip is not formed from a sub-group of parts, the largest part within it will be too big to be included in the subsequent narrower strips and is removed from the sorting. This results in a number of parts remaining unplaced when the system has tried all strip width acceptance ranges. These could be placed by an operator via an interactive system, however the new system aims to be completely automated. Another option is to use this algorithm on the remaining parts again, but this time with more relaxed strip length and width tolerances. This will result in less efficient strips being placed in the layout. To place all the parts in the list the strip tolerances would have to be repeatedly relaxed. This method is likely to result in considerable waste at the end of the layout. Thus it may be better to use this system to nest the bulk of a parts list in efficient strips and then use a second more appropriate algorithm to place the remaining parts. The 'Area Fit' algorithm is suitable for this task and is described in detail in the next section. It should be noted that the 'Strip Fit' algorithm relies on having a large range of parts from which to choose good strip combinations. The smaller the parts list, the less chance there is of placing parts without relaxing the strip tolerances. Thus the Strip Fit algorithm is most suited to large lists of parts.

When the remaining area on a sheet is too narrow to accommodate the next part, most nesting systems simply start a new sheet. This often wastes a considerable area on the last sheet. This situation would occur when the strip width, set from the largest X dimension which has not yet been tried, is larger than the X dimension of the remaining sheet area. In this situation the new algorithm puts the largest part X dimension on hold and uses the X dimension of the available sheet area to set the strip width. The acceptance tolerances are also relaxed as even a badly fitting strip in this area is better than none at all. When the next new sheet is started the system reverts to using the largest part X dimension.

So far the parts considered have not been permitted to rotate. If the parts are permitted to rotate their Y dimensions are also considered when looking for parts to fit in a defined strip width. It is possible that a part's X dimension and Y dimension conform to the strip width tolerance. In this case the larger dimension is taken as the X value; this improves the layout efficiency by leaving more strip length free for other parts to occupy. If the part is square it is kept in its original orientation. Allowing part rotation approximately doubles the dimensions available for building efficient strips and therefore should significantly increase the layout efficiency. Also, parts not utilised to form a strip using their larger dimension are evaluated a second time for a strip using their smaller dimension. However these layout improvements are at the cost of processing time.

Another variant of this system was considered during the early stages of development. This relied on splitting the sheet into grid elements and enclosing the parts to be nested within envelopes constructed from whole grid elements. This allows easier subsequent part placement as the enveloped parts can be fitted perfectly on the sheet, as shown in Figure 3.12. This grid format could also be applied to the Area Fit algorithm, but not the Strip Squeeze Algorithm.

To form a strip when part rotation is not permitted, all parts with an envelope X dimension of the same number of grid divisions would be gathered into a sub-group. The Y dimensions of the part envelopes would then be tried in various combinations

to find a set which sum to the same number of grid divisions as the sheet's Y dimension. This would be carried out for all possible grid element strip widths. The 'bridge gap' would have to be added to each part prior to enveloping, to maintain the 'whole grid element' dimensions of the envelopes.

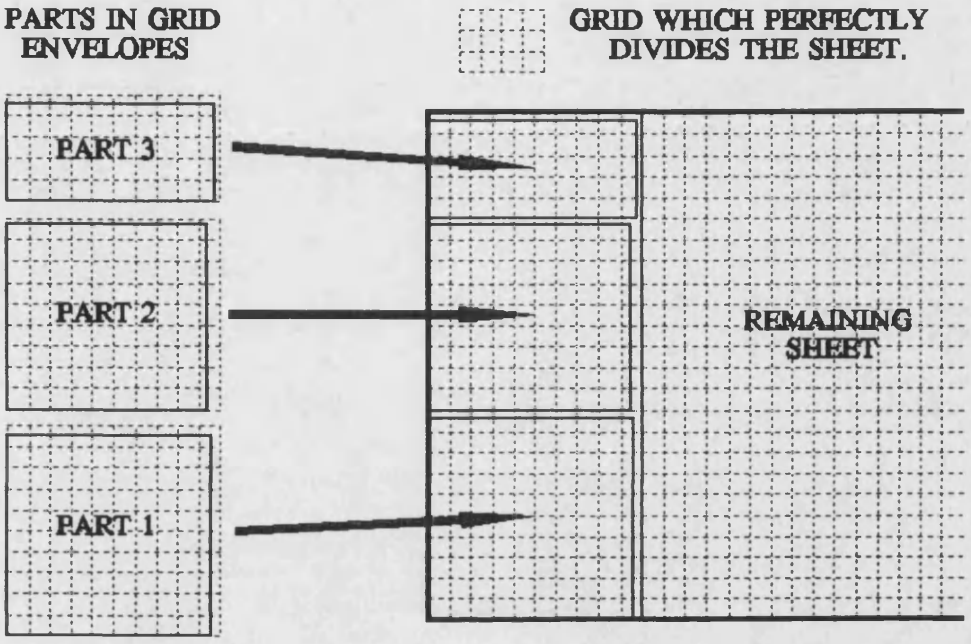


Figure 3.12.
Grid format for part placement.

The grid element size must be chosen carefully to give a good nesting result. A large grid element results in excessive waste when the parts are enveloped. A small grid element size reduces the number of parts with a common 'whole grid element' dimension, resulting in insufficient parts to form a strip. This problem is exacerbated with a small parts list. It is certain that, without the additional waste in each envelope due to the grid format, some strips formed would accommodate either a larger part or an additional part. The adopted system does not attribute waste to each envelope and this should guarantee that it always generates an equal or better layout than is produced using this grid method. This increased simplification of part placement with the grid format was considered unlikely to justify the increased waste in the generated layouts. Therefore the development of this approach was not pursued.

3.7 The Area Fit Algorithm

The Area Fit algorithm operates by identifying the largest part in the list which fits into the next space to be filled on the sheet. Initially, a new sheet is held as a single rectangular space to be filled with parts. As the layout is built up, the unoccupied sheet is represented by a number of rectangular areas around the parts. Maintaining all the space to fill on the sheet in the form of rectangular areas allows them to be represented by two dimensions and two co-ordinates. The sheet is divided around the parts to keep the largest possible single rectangular area free. The free areas remaining on a sheet are held in the order of smallest first and are nested in this order. When the parts list is exhausted any small free areas on the sheet are not retained for future use and therefore are included as waste. Thus it is preferable to nest the small free areas on the sheet first.

The Area Fit algorithm can place parts in three different ways, shown in Figure 3.13. The best type of placement is the 'Perfect Fit'. The criteria for this is when the largest part in the list, which fits within the space to nest, has an area in excess of 90% (a 'good' acceptance limit established by experimentation) of the area of the space. This is the most efficient of the three methods of part placement and is therefore the first type attempted for each area. The entire list of parts to be nested is searched for the part with the largest area which fits into the space. If this part's area is not above the 90% acceptance limit the system moves on to try the dimension fit method. Otherwise the part is placed and the next area on the sheet is considered for part placement.

The second type of part placement is the 'Dimension Fit' (Figure 3.13). This method attempts to fit a part across one side of the target area. If this is achieved the free area of the original rectangle is held and subsequently treated as an individual area to be nested. The system tries every remaining part in the bottom left of this area, calculating the difference or gap between the part and the area in both dimensions. If the part can rotate it is considered in both orientations. The smallest gap and the necessary orientation for each part are recorded. The part with the lowest associated

gap is held through the search. If the gap of the best part is less than 10% (established by experimentation) of the relevant dimension of the area to nest the part is placed. If no part is placed the system uses the third method of placing parts.

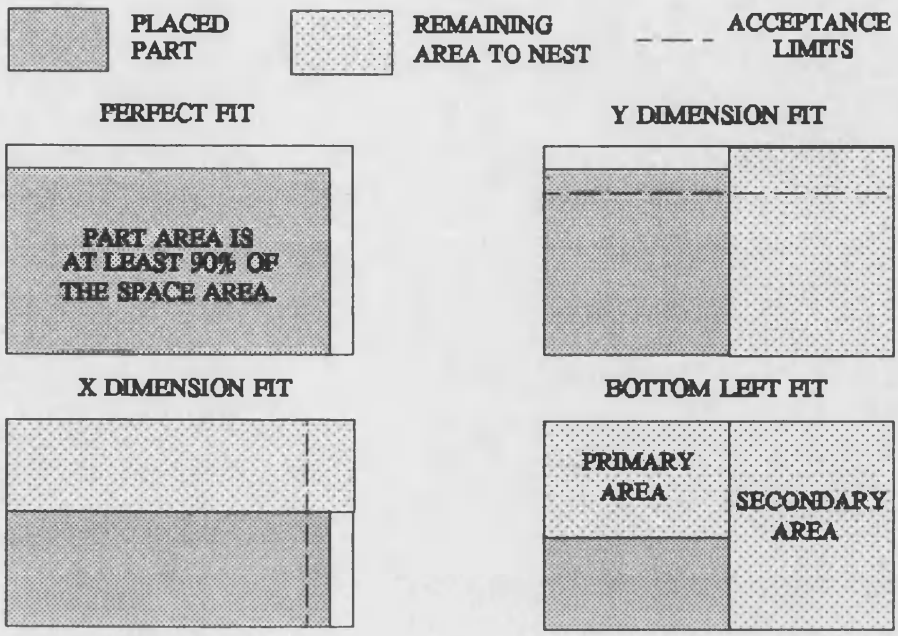


Figure 3.13.
The Area Fit algorithm's three methods of part placement.

When the space to fill is too large for any part to come close to its dimensions a part is placed in the 'bottom left' corner of the space (see example in Figure 3.13). The remaining area is divided by either a horizontal or vertical line taken from the top right corner of the placed part, to create two new rectangular spaces. If the X dimension of the space to fill is larger than its Y dimension a vertical area division line is selected for use before the part is placed (otherwise a horizontal area division line is selected). Generally, this will preserve a large secondary area (second space to fill) with a low aspect ratio. If a vertical division line is selected the part with the largest X dimension, which the space will accommodate, will be chosen for placement. Similarly, with a horizontal division line the largest Y dimension will be required. Maximising this part dimension maximises the corresponding dimension of the primary area, which in turn maximises the range of the remaining parts which this primary area can accommodate. If part rotation is permitted, the part with the largest

dimension which the space will accommodate, in the required orientation, will be chosen for placement. A step by step illustration of this algorithm's operation is given in Figure 3.14 where the Area Fit algorithm is used to complete a layout started by the Strip Fit algorithm.

Note: Pn (primary) and Sn (secondary) represent the first two positions in a 'stack' of the sheet areas to be filled, which is why P can inherit the S value. In this example the stack is never loaded with more than two areas.

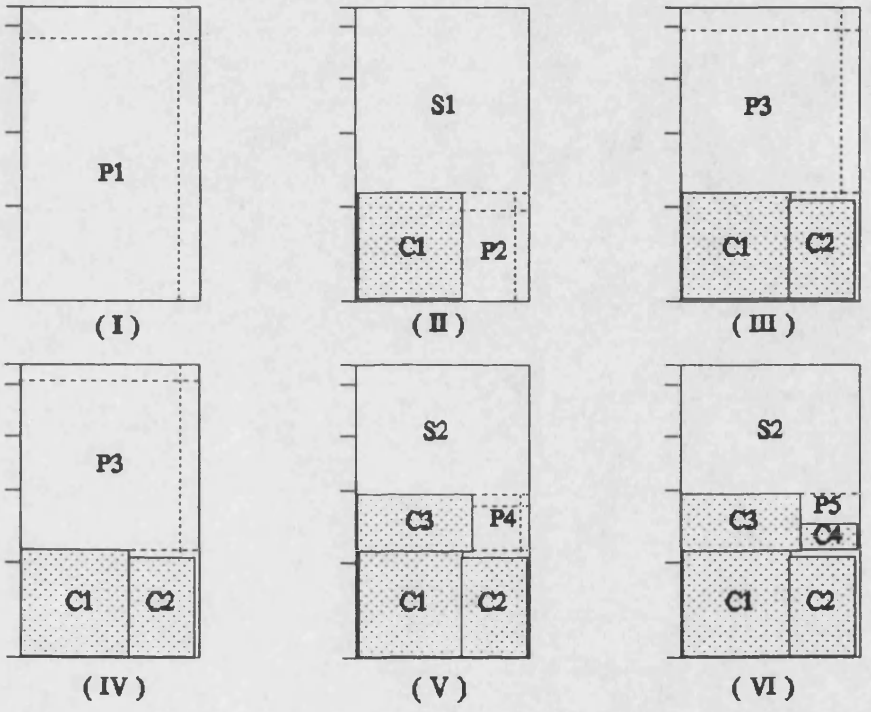


Figure 3.14.
The step by step operation of the Area Fit algorithm.

(I) Shows the area remaining P1 after the Strip Fit algorithm has been used. The dotted lines show the acceptance limits for Dimension Fit placement.

(II) As no part in the list is large enough to meet the Perfect Fit or Dimension Fit placement tolerances, the method of Bottom Left placement is used. This is to place the largest part from the list, C1, in the bottom left corner of the space. The remaining area is now divided into P2 (the primary area to nest) and S1 (the secondary area). The area could have been divided using a vertical line from the top

right corner of C1. However, it is considered better to maximise the size of the 'largest remaining area' (S1).

(III) Using the Perfect Fit method, part C2 is placed in P2. This area is now fully occupied and S1 now becomes the primary area to nest, P3.

(IV) No part can be placed in the new P3 using the Perfect Fit or Dimension Fit methods. The bottom left contingency method is therefore used to place C3.

(V) The remaining area around C3 is divided into a primary area to nest, P4 and a secondary area S2. A horizontal line from the top right corner of C3 is used as a vertical line would result in a smaller S2.

(VI) Part C4 is placed in P4 using the Dimension Fit method in the X axis. As area P4 is not completely occupied the new primary area P5 is the remaining section of P4. The secondary area S2 does not change.

A full sheet layout created by the Area Fit Algorithm alone is shown in Figure 3.15. The parts are numbered in the order in which they were placed. The letters define the method by which each part was placed. 'P' denotes a part placed by the Perfect Fit method, 'X' an X Dimension Fit, 'Y' a Y Dimension Fit and 'B' denotes a part placed by the Bottom Left method. One major advantage of this algorithm is the regular shape of the area which remains when a sheet is not completely nested. Remaining sheet sections with low aspect ratios are easy to use in the future and thus are more likely to be retained rather than being discarded as waste.

Unlike the Strip Fit algorithm, the Area Fit algorithm's performance does not drop significantly when the parts list becomes small or the parts vary considerably in size and shape. Thus, when applied individually, the Area Fit method is a more practical and robust algorithm than the Strip Fit method with repeated tolerance relaxation. The Strip Fit algorithm however, when applied to a large parts list without strip tolerance relaxation, produces a very efficient layout for many of the parts. A small list of parts will thus remain unplaced which could then be efficiently placed by the Area Fit algorithm. Thus the two algorithms are also used in series which is known as the Total Fit algorithm. It is considered that this combined system should perform better than either algorithm individually.

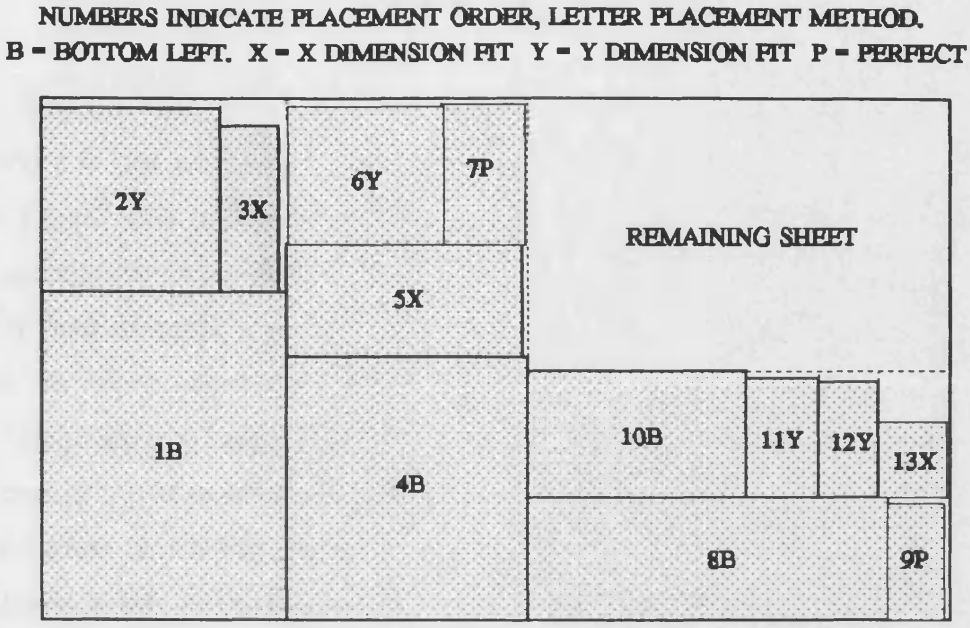


Figure 3.15.
A full sheet layout created by the Area Fit algorithm.

Another variant of this system was considered during the early stages of development. This relied on splitting the sheet into grid elements and enclosing the parts to be nested within envelopes constructed of the grid elements. As discussed at the end of Section 3.6, this allows the part envelopes to be perfectly fitted to the sheet. The justification for not pursuing the development of this approach for the Strip Fit Algorithm is also valid for its application to the Area Fit Algorithm.

3.8 Strip Squeeze Algorithm

The 'Strip Fit' algorithm generates a layout which conforms to the guillotine cut constraint ie. the parts can be cut from the sheet using straight line cuts across the sheet's span. This constraint is often not required with modern cutting methods (laser and rectangular shears) and the layout efficiency can be improved by its removal. Rather than developing a completely new algorithm to generate an unconstrained layout, an improvement algorithm could be applied to improve a constrained layout. The 'Strip Squeeze' algorithm has been developed to enhance any guillotine cut constrained layout in which the parts have been placed in strips or rows. This system can therefore be used to improve strip layouts generated by the Strip Fit algorithm developed in this research and by other placement algorithms which form strips eg. the First Fit algorithm [Coffman et al (1980)]. Permitting part rotation does not improve the performance of the Strip Squeeze algorithm as the part orientations are fixed by the placement algorithm. The step by step operation of this algorithm is shown below in Figure 3.16.

(I) Shows two strips conforming to the 'guillotine cut constraint' which have been generated by the 'Strip Fit' algorithm; this is the layout which the 'Strip Squeeze' algorithm receives. The parts are in decreasing Y dimension order, with the largest Y dimensions at the bases of the strips.
(II) The parts making up each strip are re-arranged into decreasing X dimension order, again working from the bases of the strips.
(III) The next stage of this operation is to invert the right strip and align its member parts to the right, which is effectively a 180° rotation of the entire strip.
(IV) Finally the strips can be squeezed together, reducing the area of sheet which they occupy. The smallest gap between the strips must be calculated, as this is the maximum translation of the right strip which can be performed without part overlap occurring. The top right corner of each part in the left strip is taken and the part which is opposite to it in the right strip is found. The gap between the two strips can be found by subtracting the opposing part's X dimensions from the total X dimension of the two strips. The smallest value from all the corners in the left strip is saved and

the process is repeated for the bottom left corners of the right strip. If a gap from the corners of the parts in the right strip is found to be smaller than the smallest from the left strip it becomes the smallest gap found so far. At the end of this process the right strip can be translated by the appropriate distance.

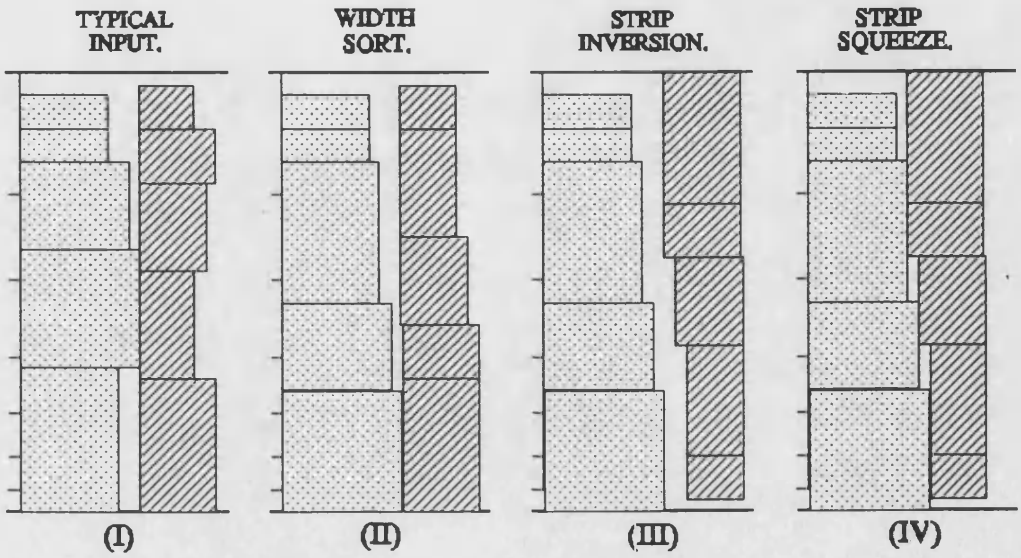


Figure 3.16.
The step by step operation of the Strip Squeeze algorithm.

If this method is applied to a single sheet layout after it has been created, it will create additional free area at the right end of the sheet. If a multiple sheet layout is treated this way the best result possible is to create a free section on each sheet which can be reused in the future. It would be possible to fill the free areas with strips from later sheets, however this would add considerable complexity to the system. A better method is to invoke the Strip Squeeze method each time two strips are placed onto a sheet, gaining the material saving immediately.

The strips could also be squeezed vertically ie. the right strip moved down as far as possible. This would reduce tool movement by a small amount, but would not improve material utilisation and therefore has not been implemented in this work.

3.9 Irregular Sheets and Bad Areas

Irregular Sheet Perimeter

Nee et al (1986) approximate an irregular shape with straight lines. Each of these lines then becomes the diagonal of a rectangle and parts may not be placed within these rectangles but may be placed within the area which these rectangles enclose. This area is in the form of a rectangular construct, shown in Figure 3.17. (Note: A 'bad area' has been removed from the original figure, shown in Section 2.2, to simplify the problem to an irregular perimeter). The initial approximation of an irregular sheet perimeter to a straight line construct is a practical simplification and is appropriate for the nesting system designed. However, the area generated for nesting by Nee et al would have to be further simplified to rectangular areas to allow nesting by the new placement algorithms. Rectangular areas S1 to S4, in Figure 3.18, could be created by taking vertical lines from the concave corners. These areas would be nested in this order, ensuring the small parts in the list are available for placement in the small areas. However, it is preferable to identify the largest single free area, therefore other division strategies were devised to achieve this.

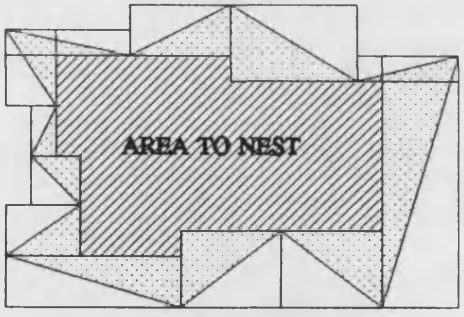


Figure 3.17.
Irregular sheet division by Nee et al.

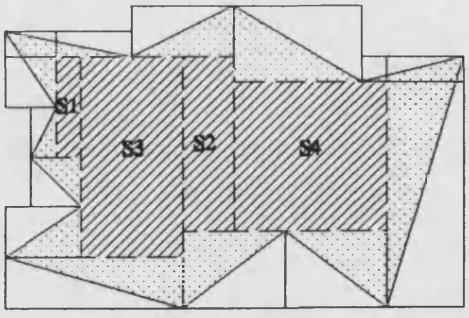


Figure 3.18.
Adaptation of construct area into rectangular areas.

As stated, to gain a good layout it seems appropriate to first maximise the size of a single rectangular segment and then the size of each subsequent segment. This ensures

that the sheet is not split into a number of similar sized segments, none of which will accommodate the largest part. This gives the system at least one large area which can be tightly packed. As most parts lists contain a few small parts, the smaller sheet division segments can be filled with these. Thus, the smallest sheet division segments should be nested first to maximise the range of small parts available.

A method for dividing an irregular sheet which is more suited to the new algorithms is shown in Figure 3.19. The concave points on the sheet perimeter, which have been circled in the diagram, have been taken as the limits of the largest principally orientated rectangle within the sheet. These are established by comparing the positions of the points either side of the point being considered. The concave points identified are then stored as limits for the appropriate rectangle side. It is possible that a single concave point may be the limit for two sides of the rectangle. If no limiting point is found for a rectangle side a contingency must used. The two rectangle sides perpendicular to the one which is undefined are projected to intersect with the perimeter and these intersections are treated as if they were convex points. The right side of this sheet in Figure 3.19 is an example of this contingency rule.

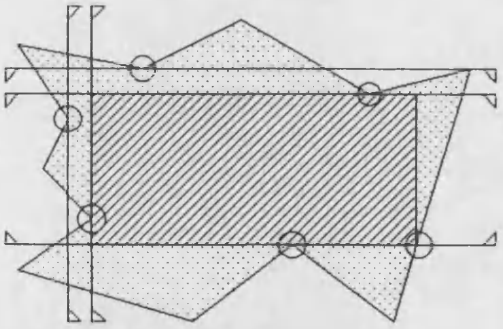


Figure 3.19.
A new irregular sheet division.

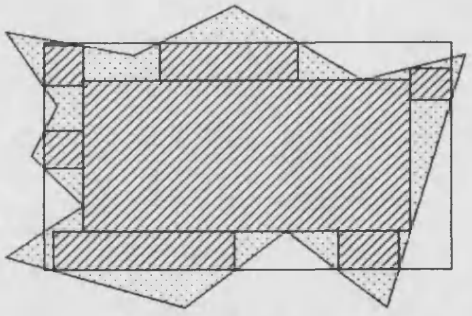


Figure 3.20.
Boundary limit method of filling perimeter spaces.

Any suitably large areas outside the primary rectangular segment are then approximated as rectangles by one of three simple methods.

The first is to project lines parallel to the primary rectangular segment's sides at a pre-determined distance from the (primary rectangles) sides. The distance would be

chosen, by examining the part dimensions, to allow a number of the smaller parts in the list to be located within it. From the intersections of the projected lines and the perimeter, lines perpendicular to the primary rectangular segment are drawn creating small rectangular areas. This is shown in Figure 3.20.

The second method is to take the mid point of each perimeter line of the irregular sheet and project horizontal and vertical construction lines from it. Additional construction lines may have to be created at intersection points with the perimeter to complete the rectangular areas. Each perimeter area will typically have two such construction rectangles within it and the larger is then taken as the rectangular segment for the area (see Figure 3.21).

The third method (Figure 3.22), and most simple, is to construct rectangular segments from the convex perimeter points discarded when forming the primary segment. However, in this example it is the most wasteful method and this is likely to be true for most sheets.

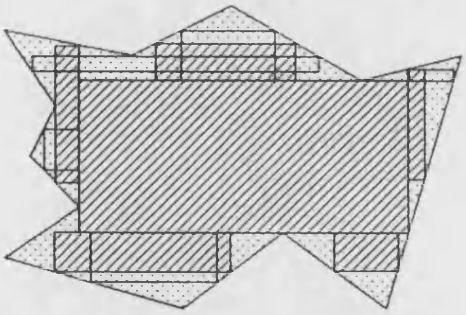


Figure 3.21.
Construct from line mid-point method of filling peripheral spaces.

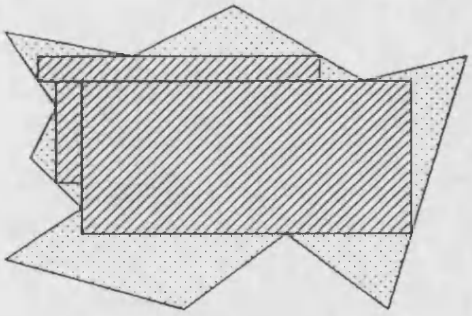


Figure 3.22.
Construction from discarded convex points to fill peripheral space.

If the sheet is permitted to rotate, a good orientation can be found to give a large single free area. This is done by sequentially taking every pair of convex perimeter points and aligning them with a principle axis. A principally orientated rectangle can then be created within the sheet's bounds according to the method described earlier. The best orientation is then selected. The five possible orientations using neighbouring convex points are given in Figure 3.23.

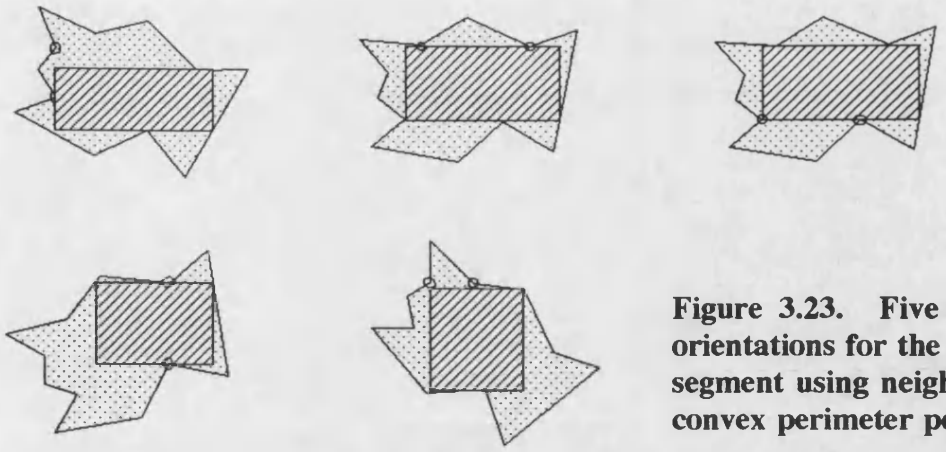


Figure 3.23. Five possible orientations for the primary segment using neighbouring convex perimeter points.

Accommodating 'Bad Areas' on the sheet

The new system designed can be adapted to nest parts onto sheets which contain 'bad areas', which must be avoided, by using the following methods. It must be noted that the more irregular the sheet becomes, the worse the algorithms will perform. If a production run is to use a number of sheets of varying integrity, the best overall efficiency will be achieved by using the more irregular sheets first, as a greater choice of parts to place will exist.

Within the system being developed a 'bad area' would have to be enclosed within a rectangular envelope. Thus its area would be represented by X and Y dimensions and its position by X and Y co-ordinates from the bottom left corner of the sheet. Figure 3.24 shows a 'bad area' with dimensions B_X,B_Y and location C_X,C_Y from the bottom left corner of a sheet with corners notated LMNO and dimensions D_X,D_Y.

The sheet must now be divided into four rectangular segments around the 'bad area'. Each of these segments can subsequently be treated by the algorithms as an individual area to nest. Example areas are shown in Figure 3.24, denoted A1-A4. The areas would be nested smallest first to ensure the greatest range of parts available for the difficult small areas. Although the 'bad area' is closer to LM, the division to

maximise the largest area is parallel to LO due to sheet LMNO not being square. The areas are ordered in the sequence in which they will be nested and are as follows.

A1	**AREA = B_X * (D_Y-C_Y-B_Y)**	**CO-ORDS = C_X , (C_Y+B_Y)**
A2	**AREA = C_X * (D_Y-C_Y)**	**CO-ORDS = 0 , C_Y**
A3	**AREA = (C_X+B_X) * C_Y**	**CO-ORDS = 0 , 0**
A4	**AREA = (D_X-C_X-B_X) * D_Y**	**CO-ORDS = (C_X+B_X) , 0**

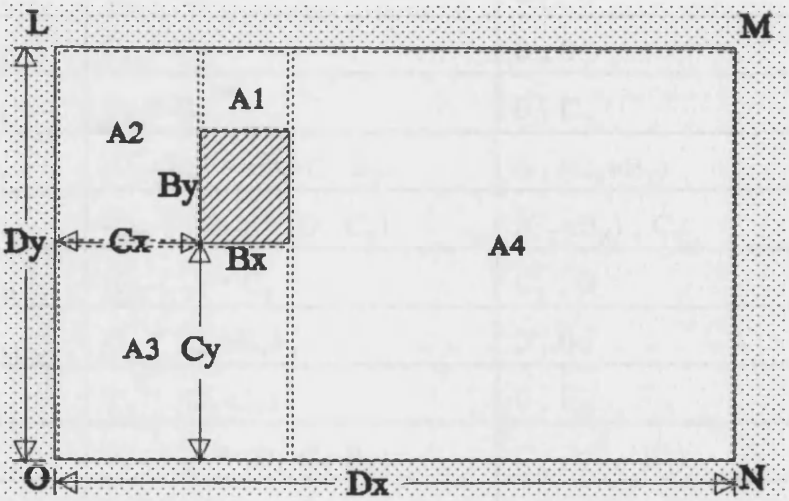


Figure 3.24.
An example 'bad area' on a sheet.

The position of 'bad areas' will differ on each sheet requiring a flexible system to find the optimum division of the remaining sheet. There are 20 possible space divisions which can be created using straight lines in this situation, shown in Figure 3.25. These spaces are referenced as S_1-S_{20} and the equations for their areas and co-ordinates are given in Table 3.1. The possible space divisions for each area to nest A_1-A_4 are given in Table 3.2 . From Table 3.2, if the largest division area around the 'bad area' is S_1 the next largest remaining area will be S_2, S_{11} or S_6. Assuming S_2 is the largest of these, the remaining two areas would be S_{14} and S_{18} in size order.

Space	Area	Co-ords
S_1	$C_X * (D_Y-C_Y-B_Y)$	$0 , (C_Y+B_Y)$
S_2	$B_X * (D_Y-C_Y-B_Y)$	$C_X , (C_Y+B_Y)$
S_3	$(D_X-C_X-B_X) * (D_Y-C_Y-B_Y)$	$(C_X+B_X) , (C_Y+B_Y)$
S_4	$(D_X-C_X-B_X) * B_Y$	$(C_X+B_X) , C_Y$
S_5	$(D_X-C_X-B_X) * C_Y$	$(C_X+B_X) , 0$
S_6	$B_X * C_Y$	$C_X , 0$
S_7	$C_X * C_Y$	$0 , 0$
S_8	$C_X * B_Y$	$0 , C_Y$
S_9	$(C_X+B_X) * (D_Y-C_Y-B_Y)$	$0 , (C_Y+B_Y)$
S_{10}	$(D_X-C_X-B_X) * (D_Y-C_Y)$	$(C_X+B_X) , C_Y$
S_{11}	$(D_X-C_X) * C_Y$	$C_X , 0$
S_{12}	$C_X * (C_Y+B_Y)$	$0 , 0$
S_{13}	$C_X * (D_Y-C_Y)$	$0 , C_Y$
S_{14}	$(D_X-C_X) * (D_Y-C_Y-B_Y)$	$C_X , (C_Y+B_Y)$
S_{15}	$(D_X-C_X-B_X) * (C_Y+B_Y)$	$(C_X+B_X) , 0$
S_{16}	$C_X * (C_Y+B_Y)$	$0 , 0$
S_{17}	$C_X * D_Y$	$0 , 0$
S_{18}	$(D_X-C_X-B_X) * D_Y$	$(C_X+B_X) , 0$
S_{19}	$D_X * (D_Y-C_Y-B_Y)$	$0 , (C_Y+B_Y)$
S_{20}	$D_X * C_Y$	$0 , 0$

Table 3.1. All possible area divisions and their equations.

A_1	A_2	A_3	A_4
S_1	S_2	S_{14}	S_{18}
		S_{18}	S_{14}
	S_{11}	S_6	S_{16}
		S_{16}	S_6
		S_{10}	S_{14}
		S_{14}	S_{10}
	S_6	S_{11}	S_{16}
		S_{16}	S_{11}
		S_7	S_{18}
		S_{18}	S_7
S_2	S_1	S_{14}	S_{18}
		S_{18}	S_{14}
	S_8	S_5	S_{14}
		S_{14}	S_5
		S_9	S_{20}
		S_{20}	S_9
	S_9	S_8	S_{20}
		S_{20}	S_8
		S_{12}	S_{18}
		S_{18}	S_{12}
S_3	S_4	S_{20}	S_{16}
		S_{16}	S_{20}
	S_{12}	S_7	S_{18}
		S_{18}	S_7
		S_{11}	S_{16}
		S_{16}	S_{11}
	S_7	S_{12}	S_{18}
		S_{18}	S_{12}
		S_8	S_{20}
		S_{20}	S_8
S_4	S_3	S_{20}	S_{16}
		S_{16}	S_{20}
	S_5	S_{10}	S_{14}
		S_{14}	S_{10}
		S_6	S_{16}
		S_{16}	S_6
	S_{10}	S_5	S_{14}
		S_{14}	S_5
		S_9	S_{20}
		S_{20}	S_9

Table 3.2. Possible Area Combinations.

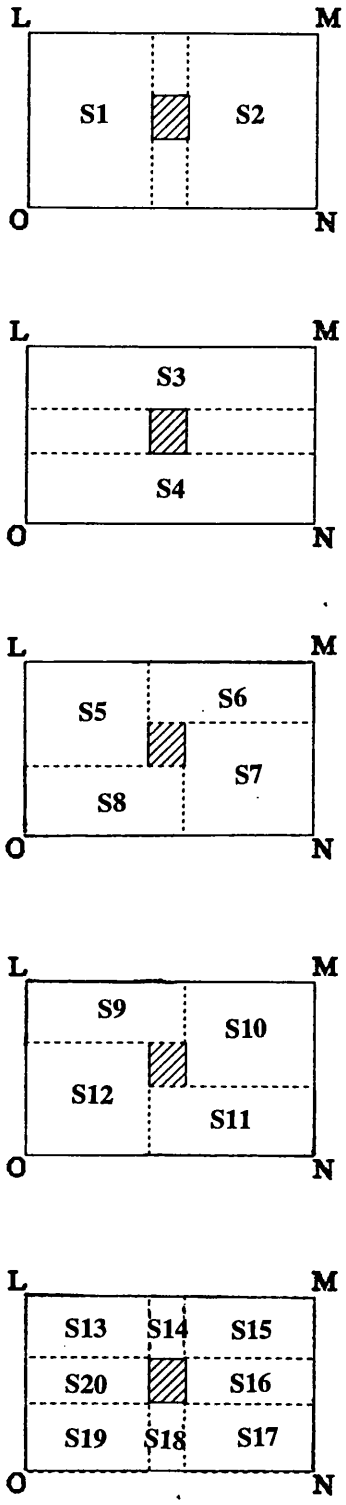


Figure 3.25. All possible straight line divisions around a single 'bad area'.

If a stock sheet contains more than one 'bad area' the sheet would still be divided in the same manner, however each part would be dealt with sequentially. The 'bad area' lying closest to the sheet periphery would take precedence and the sheet would be divided with no account of any other 'bad areas'. The sheet segment from the first division which the next 'bad area' inhabits would then be identified. This segment would then be divided around the area as if it were a whole sheet. This approach would allow a sheet with any number of 'bad areas' to have parts nested on it by the algorithms developed. The system will generate four areas around the 'bad area' regardless of its location. If the 'bad area' lies on the sheet edge one of the areas to nest will have a dimension of zero. If the 'bad area' lies in a corner this will be true of two of the areas. The system would check for this case and discard the area immediately rather than leaving it to be discarded by the nesting system. If a 'bad area' lies across a sheet segment boundary from earlier sheet division the 'bad area' must be split into two separate 'bad areas' within each segment which are then dealt with independently. If an irregular sheet contained 'bad areas' it would initially be divided into segments regardless of these areas. The sheet division segment containing the 'bad area' would then be further divided around it as described earlier.

It must be noted that the algorithms for this research have been developed specifically for regular rectangular sheets. Sheets with 'bad areas', at the edges or in the main area, will inevitably lower the efficiency of the layouts generated. However it is useful to have methods for exceptional circumstances where an irregular sheet of metal must be used. This facility also allows the system designed to be applied to planning the cutting of non-homogeneous materials, such as leather and cloth. The 'bad areas' system has two other useful applications. Firstly, the cutting method employed may require areas of the sheet to be set aside for clamping. This method would allow the material between the clamps to be utilised rather than only considering the largest single rectangular area on the sheet. Secondly, the method could be used to allow one or more parts to be given 'priority' locations on the sheet, with other parts being positioned around them.

Chapter 4

The Nesting System Developed

Only the core of the nesting system designed (Chapter 3), which carries out the placement of the rectangular envelopes, has been developed into a working system. This chapter describes the detailed structure and coding of the system developed.

4.1 Scope of the System Developed

Although a complete nesting system has been specified in Chapter 3, only the elements covered in Sections 3.6, 3.7, and 3.8 have been implemented and tested. This limits the system which is to be tested to the nesting of rectangular parts or rectangular envelopes around parts. This is the core of the total nesting system designed and the algorithms developed for this task are considered to be completely new.

The algorithms have been coded in the C language using the Borland Turbo C editor and compiler [Borland (1991a)]. Mullish and Cooper (1987), Kochan (1988) and Kernighan and Ritchie (1978) give good introductions to the C language and useful guidance on writing well structured code. Traister (1985) is of particular use when a knowledge of the Basic language exists. Borland (1991a) and Borland (1991b) give very detailed information on the use of the Turbo C editor features and the run time library functions supplied. A less detailed, but more user friendly guide to these subjects is provided by Wang and Bibb (1991). The code generated in writing the system is in excess of 10000 lines before compilation and thus is not included. However, this chapter will describe the general code structure of the system and the

detailed steps of each algorithm. The system's source code has been archived [Scott and Mileham (1995)].

The parts and the sheet sizes entered into the system have been slightly simplified to have integer dimensions in millimetres. Parts to be cut from sheet metal generally have dimensions with magnitudes of the order of tens or hundreds of millimetres. This simplification is not so crude as to result in numerous different part dimensions within a list of parts being approximated to the same value. Therefore this simplification of the nesting problem should not have any derogatory effect on the validity of the layouts produced and their associated waste.

In a practical situation the bridge gap would be established on the basis of cutting process, material type and the thickness of the sheet to be used. For the purposes of testing the algorithms against each other, varying the bridge gap is unnecessary and it will therefore be held at a constant value of 10mm. This is applied to the sheet perimeter as well as the parts. In a practical nesting situation a number of parts may be tagged as having to hold a fixed orientation on the sheet. In studying the performance of each algorithm it is necessary to quantify any deterioration in layout efficiency which is the result of parts being constrained to a fixed orientation. Evaluating this effect is simplest if either all the parts in the list are held to a fixed orientation or all the parts are allowed to be rotated. Thus, as individual part tagging for fixed orientation is unnecessary in the study of the algorithms performance, this facility has not been included in the system developed.

In addition to the three rectangular part nesting algorithms introduced in Chapter 3 the 'Next Fit' and 'First Fit' algorithms [Coffman et al (1980)], which have been reviewed in Section 2.2, have been coded in the system. These two algorithms are well established and therefore provide good 'benchmarks' for the evaluation of the new algorithms. The Strip Fit and Area Fit algorithms will be tested both individually and in series, with the use of the two algorithms in series known as the 'Total Fit' system. Thus five individual algorithms have been written. The Strip Squeeze algorithm is used to improve the layouts generated by other algorithms, however as the Area Fit

algorithm does not produce strips of parts the Strip Squeeze algorithm cannot be applied to it. Thus there will be an additional four algorithm variants due to the application of the Strip Squeeze algorithm. However, as all the algorithms and variants mentioned so far can nest parts with their orientations fixed or take advantage of part rotation being permitted, there will be 18 different variations of system and layout constraint to be tested. These are listed in Table 4.1. The layouts created by these algorithms are reviewed in Chapter 5 and the detailed testing of the algorithms is described in Chapter 6.

ALGORITHM	PART ROTATION PERMITTED	GUILLOTINE CUT CONSTRAINT IMPOSED (IF NOT STRIP SQUEEZE USED)
NEXT FIT	N	Y
NEXT FIT	N	N
NEXT FIT	Y	Y
NEXT FIT	Y	N
FIRST FIT	N	Y
FIRST FIT	N	N
FIRST FIT	Y	Y
FIRST FIT	Y	N
STRIP FIT	N	Y
STRIP FIT	N	N
STRIP FIT	Y	Y
STRIP FIT	Y	N
AREA FIT	N	Y & N
AREA FIT	Y	Y & N
TOTAL FIT	N	Y
TOTAL FIT	N	N
TOTAL FIT	Y	Y
TOTAL FIT	Y	N

Table 4.1. The 18 algorithm variants tested.

4.2 Coding and Structure of the System

As stated in Section 4.1 the system has been coded in the C language. The parts and the free areas on the sheet are held in linked lists of dedicated structures. Arrays of data are avoided due to the need for pre-determining their size, which reserves unnecessary memory and the large amount of processing time required to sort array data. Figure 4.1 shows the declaration of the structure for a nested part.

```
struct nested_part
{
      int id_num;
      int x_dim;
      int y_dim;
      int x_pos;
      int y_pos;
      struct nested_part *next_part;
};
```

Figure 4.1.
The structure to contain a nested part.

If a pointer to a structure of this type were called **this_part**, the X dimension of the part **x_dim** would be set to zero by **this_part->x_dim = 0;**. The last piece of data in the structure is specified as a pointer to the same type of structure, which can be set to another structure containing the information for a nested part.

A chain of these structures can be formed and this is known as a linked list. The linked lists used in the system are all closed ie. the pointer of the last data structure holds the address of the first data structure, as shown in Figure 4.2. The alternative to this would be to terminate the list with a NULL pointer or a 'dummy' structure. All the lists used in this system have single links, with each structure containing one pointer to the next structure in the list. In a double linked list each structure contains a second pointer which would hold the address of its preceding structure.

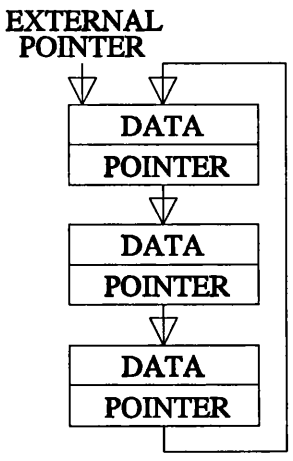


Figure 4.2.
A closed linked list.

For every data structure used a function is required to allocate the memory space it occupies. In this system all memory is allocated using the 'calloc' command (clear and allocate memory) which sets all of the data values in the new structure to zero. For the structures which are used in linked lists delete functions have also been written, which will erase a closed linked list of any length. This uses the 'free' command (free allocated memory) which can also be applied to any individual structure. If an individual structure is to be erased from a linked list, the pointer of the preceding structure must be set to the address held by the pointer of structure to be deleted. This is shown in Figure 4.3. If this is not done the addresses of all the structures after the one deleted will be lost. The incomplete linked list formed will cause the program to crash.

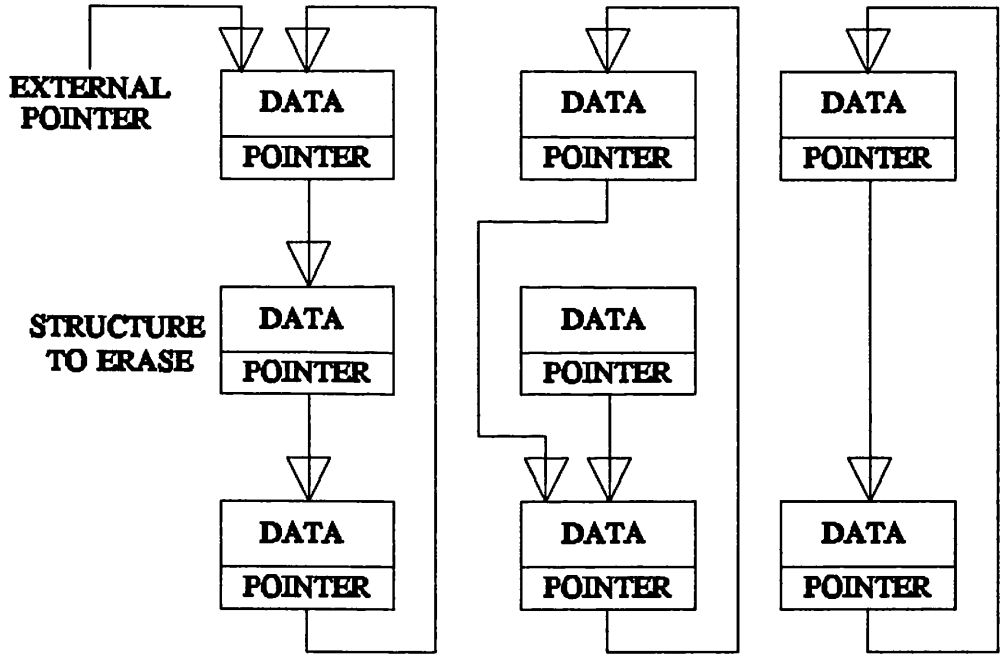


Figure 4.3.
Deleting a part in a linked list.

Similarly, when a structure is added to a linked list its pointer must be set to the address of the structure which will follow it in the list. At the time this would be held by the pointer of the structure which will precede it in the list, which in turn must be altered to the address of the new structure. If the structure which is first in a list is to be changed or deleted the external pointer to the list must be altered accordingly.

The system is written in a highly structured format allowing many functions to be shared. Only values which are required throughout the system, such as the bridge gap value or the flags which indicate the nesting constraints imposed, are made 'global' variables. All other data is passed directly from the parent function to the function it is calling. The code in Figure 4.4 is a function to rotate a part, held in the example structure introduced earlier in this section, through 90°. This can be done by swapping their X and Y dimensions. The part is called **this_part** and the function is called **rot_part**. The **rot_part** function is called from the parent function as follows: **this_part = rot_part(this_part);** In this function, shown in Figure 4.4, the type of data being passed to the function has to be specified, as does the type of data which will be returned to the parent function. The function call simply names the actual pieces of data. Only a structure of the correct type can be passed and no further data can be accessed from the parent function. The integer **hold_dimension** will only exist while the **rot_part** function is in use.

```
struct nested_part *rot_part(struct nested_part *this_part)
{
      int  hold_dimension;

      hold_dimension = this_part->x_dim;
      this_part->x_dim = this_part->y_dim;
      this_part->y_dim = hold_dimension;

      return(this_part);
}
```

Figure 4.4.
An example of a function.

The parts to be nested are held in a linked list of structures similar to the output structure introduced earlier in this section. However, the '**x_pos**' and '**y_pos**' elements, which hold the co-ordinates of the part's position on the sheet, are not included in the input structure. The list of parts to nest is read into the nesting system from a data file. Each part in the file occupies one line, which contains the integer values of its identification number, its X dimension and its Y dimension. The file is terminated by a de-limiting part with zero for all values. An example of this input file format for a list of four parts is shown in Figure 4.5. The output list of parts is also written to file in this way, with the addition of two further integer values on each line

specifying the co-ordinates of the part's bottom left corner (from the bottom left corner of the sheet). The start of a new sheet would be represented by an part identification number and co-ordinates set to zero. The X and Y dimensions would contain the values for the sheet, as shown on the first line of Figure 4.6.

```
1     345   658
2     300   200
3     455   800
4     150   450
0       0     0
```

Figure 4.5.
Input file format.

```
0     2000 1000    0     0
3      455  800   10    10
4      450  150   10   820
1      345  658  475    10
2      300  200  475   678
0        0    0    0     0
```

Figure 4.6.
Output file format.

Any parts which are too large to be placed on the sheet are removed from the list of parts to nest and placed in a separate list. As well as potentially reducing the size of the list to be handled by each algorithm, this approach allows the algorithms to simply keep nesting until no more parts remain in the list. The function to check part size only considers parts in their current orientation. If the parts are permitted to rotate, they are orientated to have their longer dimension in the same axis as the sheet's longer dimension. This ensures that only part's which cannot possibly be placed on the sheet in their primary orientations are removed from the list to nest. As this is carried out for all the algorithms it is not included in their individual descriptions.

Most of the algorithms require the input parts list to be sorted into a defined order, for instance by decreasing part area. This is a relatively quick process with linked lists as only the pointer of each member is altered, not the memory location of the data. One method is used to do this, regardless of the criteria of the sort. Initially the entire list is searched for the part with the largest area and the external pointer to the list would be set to this part's address. The rest of the list is then searched for the next largest part and pointers altered to place the part found as the next part in the list. This process is repeated until all parts are placed in the required sequence.

4.3 The Pseudocode for each algorithm

Figures 4.7 to 4.12 show the pseudocode representations of the nesting algorithms which have been written. The loop and condition statements are given in the format used in C code, but the conditions and operations have been replaced with non-mathematical descriptions. All the operations within the pseudocode are referenced with an alpha-numeric code. This code refers to sections within Appendix A, in which the sub-routines are listed in alphabetical order and described in detail. The letter in the code refers to the algorithm to which the sub-routine is dedicated: (F*) - First Fit, (S*) Strip Fit, (A*) - Area Fit and (Q*) Strip SQueezing. There are also a number of (G*) General sub-routines which are common to more than one algorithm. Next Fit algorithm does not include any dedicated sub-routines. The input and sorting of the parts, the method of waste analysis and the graphical representation of the layout produced have all been omitted from the figures for clarity.

The Next Fit Algorithm

This algorithm, introduced by Coffman et al (1980) (see Section 2.2), was originally applied to the bin packing problem, which requires the creation of horizontal rows of parts within a bin of a specified width and an unlimited depth. To apply this algorithm to the sheet nesting problem with minimal alteration, the part rows or strips are created across the smaller sheet dimension. The algorithm places parts in a given sequence, which for the bin packing was in decreasing (technically non-increasing) height order. To follow this convention, if the Y dimension of the sheet is smaller than the X dimension, the parts are placed in decreasing X dimension order. This part sorting occurs before part placement commences. If part rotation is permitted, the parts are orientated with their largest dimension on the X axis. Any parts which are too large to be placed on the sheet are removed from the parts list at this stage. The algorithm has also been adapted to recognise when the sheet is full and a new sheet must be started.

This is the simplest algorithm included in the overall system to test. As the sheet is initially empty, the first part can be placed immediately and removed from the list, giving a new top part in the list. Following this the program repeats the main 'while' loop, trying to place the top part, until the input list is exhausted. As the Next Fit algorithm generates strips of parts, the pseudocode contains the necessary calls to use the Strip Squeeze algorithm to improve the layout.

```
START A NEW SHEET (G1)
if (no guillotine constraint){SET FOR SQUEEZE (G2)}
PLACE PART (G3)
STRIP SPLIT THE PRIMARY AREA (G5)
while (parts remain to nest){
    if (top part in list fits the primary area){
        PLACE PART (G3)
        if (no secondary area [ie. starting a new strip]){
            STRIP SPLIT THE PRIMARY AREA (G5)
        } else{
            REDUCE THE PRIMARY AREA BY STRIP PLACEMENT (G6)
        }
    } else [ie. part is too big for primary area]{
        if (secondary area to nest remains){
            MAKE SECONDARY AREA PRIMARY (G4)
            if (no guillotine constraint and two strips placed){
                APPLY THE STRIP SQUEEZE ALGORITHM
            }
        } else [ie. sheet is full]{
            START A NEW SHEET (G1)
            if (no guillotine constraint){SET FOR SQUEEZE (G2)}
        }
    }
}
if (no guillotine constraint and two strips placed){
    APPLY THE STRIP SQUEEZE ALGORITHM
}
```

Figure 4.7.
Pseudocode for the Next Fit algorithm.

```
START A NEW SHEET (G1)
if (no guillotine constraint){SET FOR SQUEEZE (G2)}
PLACE PART (G3)
STRIP SPLIT THE PRIMARY AREA (G5)
while (parts remain to nest){
    if (an end area exists){
        FIND AN END AREA FOR THE NEXT PART (F1)
        if (an end area is found){
            PLACE PART (G3)
            ADJUST END AREA AFTER PART PLACEMENT (F2)
        }
    }
    if (part is not placed in an end area){
        if (top part in list fits the primary area){
            PLACE PART (G3)
            if (no secondary area [ie. starting a new strip]){
                STRIP SPLIT THE PRIMARY AREA (G5)
            } else{
                REDUCE THE PRIMARY AREA BY STRIP PLACEMENT (G6)
            }
        } else [ie. part is too big for primary area]{
            if (secondary area to nest remains){
                MAKE PRIMARY AREA AN END AREA (F3)
                MAKE SECONDARY AREA PRIMARY (G4)
                if (no guillotine constraint and two strips placed){
                    APPLY THE STRIP SQUEEZE ALGORITHM
                }
            } else [ie. sheet is full]{
                START A NEW SHEET (G1)
                if (no guillotine constraint){SET FOR SQUEEZE (G2)}
            }
        }
    }
}
if (no guillotine constraint and two strips placed){
    APPLY THE STRIP SQUEEZE ALGORITHM
}
```

Figure 4.8.
Pseudocode for the First Fit algorithm.

The First Fit Algorithm

This algorithm, also by Coffman et al (1980), is an improvement of the Next Fit algorithm. When the top part in the list cannot be placed in the area remaining at the end of the current strip, the Next Fit algorithm discards the area. However, it may be possible to place a part which is further down the list within this area. The First Fit algorithm retains these 'end areas' and attempts to place parts within them. The pseudocode for the First Fit algorithm is shown in Figure 4.8. If a comparison is made with the pseudocode for the Next Fit algorithm, it can be seen that the 'end areas' section has been added into the main loop before the normal placement section. The placing of a part within an 'end area' has priority over the normal form of part placement. This is because the 'end areas' have been written off as waste in the Next Fit algorithm, and therefore placing parts within them can only improve the layout efficiency. No 'end area' will exist until the first strip has been formed, at which point an additional function (F3) will create the first 'end area'.

The Strip Fit Algorithm.

The Strip Fit algorithm is a novel approach developed in this work and is the most complicated of the five algorithms coded in the overall nesting system. The algorithm is described in general terms in Section 3.6. The pseudocode representation given in Figure 4.9 is for the algorithm without strip tolerance relaxation. In this format the algorithm would rarely nest all the parts in a list and a second algorithm would be required to 'tidy up' the remaining parts. The Strip Fit algorithm was originally intended to operate in this manner and the **Total Fit Algorithm** uses it in this form, with the Area Fit algorithm being employed afterwards to nest any unplaced parts. The Total Fit algorithm is not shown in pseudocode as it simply consists of the Strip Fit code followed by the Area Fit code. When the Strip Fit algorithm is used on its own the strip acceptance tolerances must be repeatedly relaxed to ensure that all the parts within the list are placed. This will be discussed in detail later in this section.

The algorithm attempts to form groups of parts which form strips within set acceptance tolerances which ensure that only strips with very little associated waste are created. Generally the strip widths are derived from the dimensions of the parts in the list. However if the remaining area is too narrow to accommodate the strip width intended, the algorithm defaults to fit a new strip group on the basis of the remaining area width ie. the first 'if' condition in Figure 4.9 is entered. The new strip width limits are set in sub-routine (S3) and later cancelled in (S8). Initially the formation of a strip within the acceptance tolerances is attempted as with the normal strip fit method, sub-routines (S4) and (S5). However, if this cannot be achieved a contingency method is employed. A sub-group is formed which contains any parts narrower than the area. From these a strip group, which does not exceed the strip length constraint, is formed on the basis of widest first, by sub-routines (S6) and (S7). Inefficient part placement can be accepted in this area, as the whole area is otherwise accepted as waste when a new sheet is started.

If the remaining area is large enough to accommodate the intended strip width the corresponding 'else' condition is entered and a strip group formed again using the conventional sub-routines (S4) and (S5). Regardless of the method of strip group formation, the parts in the strip are removed from the parts list by the same method, sub-routine (S9), and placed by methods which are general to the placing of parts in strips. The strip width limits are altered every time the program completes a loop.

To use the Strip Fit algorithm alone to nest a complete list of parts, the constraints on the efficiency of the strips produced must be repeatedly relaxed. Figure 4.10 shows the adaption of the algorithm for this purpose. The code omitted from the inner 'do...while' loop in Figure 4.10 is identical to the code within the 'do...while' loop in Figure 4.9. Each time the strip acceptance tolerances are to be relaxed the length and width tolerances are alternately relaxed.

```
START A NEW SHEET (G1)
if (no guillotine constraint){SET FOR SQUEEZE (G2)}
SET NEW STRIP WIDTH LIMITS (S2)
do{
    if (intended strip width is wider than remaining area ie. end of sheet){
        SET END STRIP WIDTH LIMITS (S3)
        COLLECT ALL THE PARTS WITHIN THE WIDTH LIMITS (S4)
        TRY TO FORM A STRIP GROUP (S5)
        if (no strip group formed){
            COLLECT ALL PARTS NARROWER THAN THE AREA (S6)
            FORM AN END STRIP GROUP (S7)
        }
        RE-SET NORMAL STRIP WIDTH LIMITS (S8)
    } else [ie. normal strip width limits apply] {
        COLLECT ALL THE PARTS WITHIN THE WIDTH LIMITS (S4)
        TRY TO FORM A STRIP GROUP (S5)
    }
    if (a strip group has been formed){
        REMOVE STRIP GROUP PARTS FROM THE INPUT LIST (S9)
        if (part rotation is required){ROTATE THE NECESSARY PARTS (S10)}
        STRIP SPLIT THE PRIMARY AREA (G5)
        while (parts remain in the strip group){
            PLACE PART (G3)
            REDUCE THE PRIMARY AREA BY STRIP PLACEMENT (G6)
        }
        if (no guillotine constraint and two strips placed){
            APPLY THE STRIP SQUEEZE ALGORITHM
        }
    }
    if (end strip placement attempted ie. sheet is full){
        START A NEW SHEET (G1)
        if (no guillotine constraint){SET FOR SQUEEZE (G2)}
    }
    SET NEW STRIP WIDTH LIMITS (S2)
} while (untried strip widths and parts to nest remain)
```

Figure 4.9.
Pseudocode for the Strip Fit algorithm.

```
START A NEW SHEET (G1)
if (no guillotine constraint){SET FOR SQUEEZE (G2)}
SET NEW STRIP WIDTH LIMITS (S2)
do{
    do{
        APPLY THE STRIP FIT CODE WITHIN THE
        'DO...WHILE' LOOP IN FIGURE 4.9.
  } while (untried strip widths remain)
    RELAX THE STRIP ACCEPTANCE TOLERANCES (S1)
} while (parts in the list remain)
```

Figure 4.10.
Pseudocode for the Strip Fit algorithm adapted to nest complete lists of parts.

The Area Fit Algorithm.

The pseudocode for the Area Fit algorithm is shown in Figure 4.11. The program loops every time a part is placed or an area which is too small to accommodate any remaining part is discarded. Initially the list is searched, by sub-routine (A1), to find a part which fits the current area 'perfectly' ie. within 90% of the area. If no 'perfect fit' part is found, the list is searched for a part which closely fits the area in one dimension, sub-routine (A2). If a part has been found at this stage it is placed and the program bypasses the remaining routines to complete a loop. If a part has not been found the list is searched for the largest part to fit the current area, sub-routine (A4). If a part is found it is placed and the program again bypasses the remaining routines. If no part is found this area is discarded ie. accepted as waste and the next area is considered in the next program loop. If no 'next area' exists the current sheet has been completed and a new sheet must be started.

```
START A NEW SHEET (G1)
do{
    SEARCH FOR A PERFECT FIT PART (A1)
    if (no perfect fit part found){
        SEARCH FOR A DIMENSION FIT PART (A2)
    }
    if (perfect fit or dimension fit part found){
    PLACE PART (G3)
    ADJUST PRIMARY AREA FOR DIMENSION FIT (A3)
    }
    if (no part yet found to fit the area){
        SEARCH FOR A BOTTOM LEFT FIT PART (A4)
        if (bottom left fit part found){
            PLACE PART (G3)
            BOTTOM LEFT SPLIT PRIMARY AREA (A5)
        }
    }
    if (no part would fit the area){
        if (secondary area to nest remains){
            MAKE SECONDARY AREA PRIMARY (G4)
        } else [ie. sheet is full]{
            START A NEW SHEET (G1)
        }
    }
} while (parts remain to nest)
```

Figure 4.11.
Pseudocode for the Area Fit algorithm.

The Strip Squeezing Algorithm.

The pseudocode for all previous algorithms, except the Area Fit, contain calls to this algorithm, the purpose of which is to improve an existing strip layout rather than creating an original layout. For this reason the algorithm's pseudocode, shown in Figure 4.12, consists of a series of steps rather than a loop. The two strips to squeeze are removed from the list of nested parts, separated and then sorted into decreasing X

dimension order. The first strip's parts are re-positioned from the bottom left of the first strip's area and the second strip's parts are re-positioned from the top right of the second strip's area. The smallest gap between the strips is the lowest value found by sub-routines (Q5) and (Q6). The parts in the second strip are translated left by this gap and the remaining area(s) on the sheet altered accordingly. The strips are then linked back into the list of nested parts.

```
SEPARATE THE STRIPS FROM THE LIST OF NESTED PARTS (Q1)
SORT STRIP 1 TO X DIMENSION ORDER (Q2)
for (all parts in strip 1){
    PLACE PARTS FROM THE BOTTOM LEFT (Q3)
}
    SORT STRIP 2 TO X DIMENSION ORDER (Q2)
for (all parts in strip 2){
    PLACE PARTS FROM THE TOP RIGHT (Q4)
}
for (all parts in strip 1){
    FIND GAPS AT TOP RIGHT PART CORNERS (Q5)
}
for (all parts in strip 2){
    FIND GAPS AT BOTTOM LEFT PART CORNERS (Q6)
}
if (smallest gap found is greater than zero){
    LEFT TRANSLATE STRIP 2 PARTS BY THE GAP (Q7)
    CHANGE THE REMAINING AREA (Q8)
    if (first fit algorithm being used){
        CHANGE THE APPROPRIATE END AREAS (Q9)
    }
}
RETURN THE STRIPS TO THE LIST OF NESTED PARTS (Q10)
```

Figure 4.12.
Pseudocode for the Strip Squeeze algorithm.

4.4 Calculation of Layout Waste

The efficiency of a layout can be calculated by the following equation:

Efficiency (%) = (Area of the Parts / Occupied Sheet Area) * 100

It is considered to be desirable for the efficiency statistic to only reflect the effectiveness of the algorithm, therefore waste which is beyond the control of the algorithm, such as the Bridge Gap, is not included. This is achieved by adding the appropriate Bridge Gap to the dimensions of the parts when calculating the 'Area of the Parts'. As a Bridge Gap is placed around the sheet perimeter, the appropriate value is also subtracted from the dimensions of the sheet in the calculation of the 'Occupied Sheet Area'. Also, as all the parts to be nested are rectangles there is no 'internal' or 'envelope' waste (see Section 1.9). No account is made of any clamping waste incurred with the cutting process.

The 'Occupied Sheet Area' is the total area of the sheets used, minus the area of any remaining spaces on the last sheet which can be retained for later use. The most generous approach is to consider any remaining spaces on the sheet, into which the algorithm would continue to place parts, as being be reusable, regardless of their size. Of the algorithms in the new nesting system only the Area Fit may hold more than two spaces on a sheet. With this algorithm any spaces other than the largest two are exceptionally small and therefore impractical for retention and reuse. Thus in the new system the 'Actual Remaining Area' is the sum of the areas of the two largest spaces yet to be filled on the last sheet. These areas are identified by grey shading in the example layouts given in Chapter 5. The 'Actual Remaining Area' value is used to calculate the **Actual Efficiency** of the layout.

$$\textbf{Actual Efficiency} = \frac{\textbf{Area of the Parts}}{\textbf{Total Sheet Area - Actual Remaining Area}} * 100$$

In a practical situation only suitably large remaining spaces on the sheet would be retained for subsequent nesting. Generally this would be the largest single remaining rectangular space, providing it exceeded some minimum size limits. The 'Practical Remaining Area' which is retained by this system requires an area in excess of 180000mm^2 and neither dimension to be less than 300mm. Thus a second performance statistic, the **Practical Efficiency**, is calculated using this 'Practical Remaining Area'.

$$\textbf{Practical Efficiency} = \frac{\textbf{Area of the Parts}}{\textbf{Total Sheet Area - Practical Remaining Area}} * 100$$

The evaluation of remaining areas strictly adheres to the sheet divisions made by the nesting algorithm. There are situations where the remaining space at the end of the last sheet could be divided in a different manner to maximise the difference between the areas of the two remaining areas. This would increase the area of the larger remaining space and in turn the Practical Remaining Area, which would result in a better Practical Efficiency. Figures 5.12 to 5.15 in Chapter 5 are good examples of this situation. In all cases a horizontal division of the remaining area from the corner of the last part placed would have increased the area of the largest single remaining space, giving a better Practical Efficiency value. However, such a division is contrary to the operation of the algorithm and therefore is not permitted.

4.5 Generating Parts Lists

Random Method

In order to test the nesting algorithms it is necessary to generate some lists of parts to be nested. Creating all of the required lists manually would be tiresome, so two basic part generation systems have been developed within the new system. The first is a random generation system to create a list of parts within specified bounds. Into this system the user enters the largest and smallest permissable part dimensions and the number of parts required. The pseudocode for random part generation is shown in Figure 4.13. The dimension limits are held as 'MAX_DIM' and 'MIN_DIM'. A random value between zero and one is produced by 'RANDOM', which is then translated into an actual integer X or Y value.

```
GET THE PARTS LIST BOUNDS
for (the number of parts required) {
        PART X DIM = MIN_DIM + RANDOM * (MAX_DIM - MIN_DIM)
        PART Y DIM = MIN_DIM + RANDOM * (MAX_DIM - MIN_DIM)
        ADD THE PART TO THE LIST
}
WRITE THE PARTS LIST TO FILE
```

Figure 4.13.
Pseudocode for Random Parts list generation.

Manual Method

The new system also has a facility to create a parts list manually by entering the two part dimensions and the 'number off' of that type of part required in the list. In both systems the parts are automatically allocated an incremental part identification number. Both the Random and Manual methods display the values of the last 25 parts generated and the critical parts list statistics, shown in Figure 4.17, Section 4.6.

Biased Aspect Ratio Method

The parts within a list can be biased towards a particular aspect ratio by setting different ranges for the two part dimensions. As the two different dimension ranges will produce different mean values, the average aspect ratio of the parts can be forced to a higher value. This value is not calculated or predicted with any accuracy within the system, so a process of trial and error has been adopted to give the required average aspect ratio of the parts.

Fixed Aspect Ratio Method

In order to test the effect of part aspect ratio on nesting efficiency it is preferable to fix the average part size; this will allow differences in layout efficiency to be more easily attributed to the part aspect ratio. This has been achieved by creating four parts, each with a different aspect ratio, from 36 square grid elements. These parts are shown in Figure 4.14. By randomly generating different sized grid elements, four lists of parts with defined aspect ratio's have been created. To prevent impractically small or large parts being created upper and lower bounds on the part's dimensions are set. This considerably constrains the dimension range of the grid element as the dimension bounds must be commonly applicable to all four sets of parts. The maximum dimension of the square grid element will be $^1/_{18}$th of the maximum part dimension and the minimum dimension of the grid element will be $^1/_2$ the minimum part dimension, to account for the part with the 1 : 9 aspect ratio. With maximum and minimum part dimensions set at 900mm and 100mm the only acceptable dimension of the grid element is 50mm and therefore all the parts in each list would be identical. Thus the part dimension ranges need to be chosen carefully.

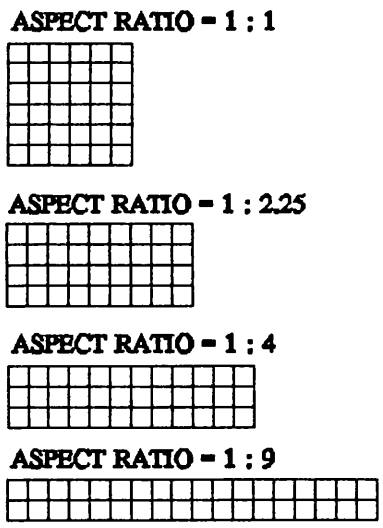


Figure 4.14.
Fixed Aspect Ratio Parts.

Fixed Area Method

This method allows the average part area to be held to the same value over a number of parts lists. This is achieved by repeatedly creating groups of parts, by randomly dividing a pre-determined rectangular area into a number of (in this case 10) segments. A parts list, with its size in multiples of 10, can be created by combining the appropriate number of these groups. Regardless of the number of times this is repeated, if the rectangular area is the same each group of parts created will have the same sum area and therefore the same average area.

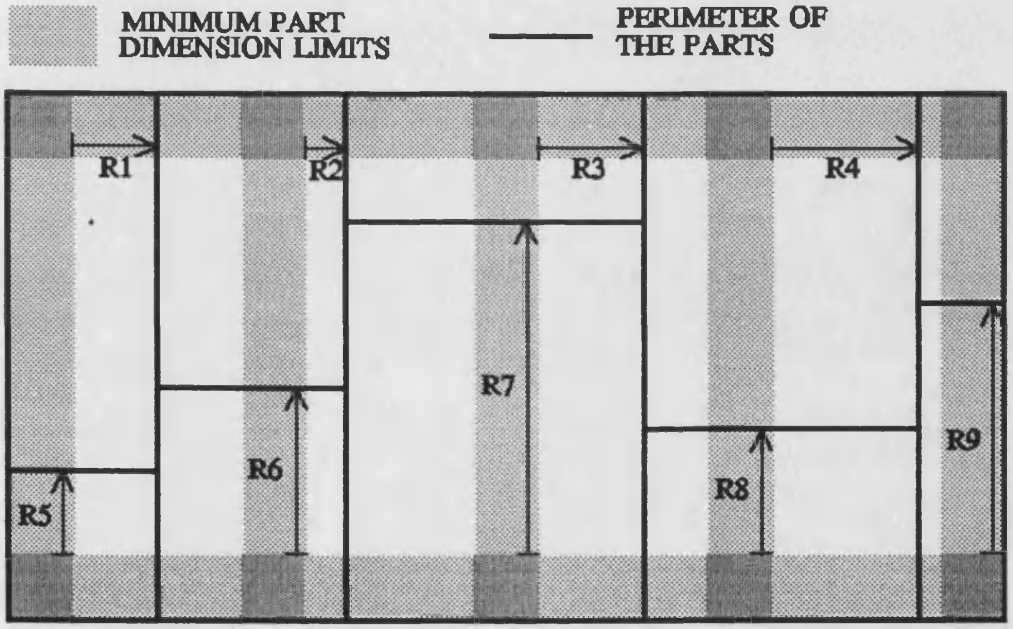


Figure 4.15.
Method of generating parts with a defined average area.
(R numbers indicate the nine random values used to divide the area)

The area to divide is shown in Figure 4.15. As with the other methods of part generation, a minimum part dimension is set to prevent impractically small parts from being created. Four vertical divisions are made in free areas between the minimum part dimension limits to split the original area into five vertical strips. The position of each split line is set randomly within the X dimension of the free area, as shown

by R1 to R4 in Figure 4.15. Each of the five strips are then split by horizontal dividing lines, the positions of which are also set randomly, R5 to R9 in Figure 4.15. This results in 10 segments, which are used as parts. Again a minimum part dimension is used to constrain the width of each strip and the dividing line to within pre-defined limits. Of the ten parts generated there are five pairs of parts which share an X dimension. This is changed by rotating the bottom five parts through 90°.

This method is used to create the parts lists for evaluating the affect of the 'sheet area to average part area' ratio on nesting efficiency. To allow the sheet areas to be set at convenient incremental sizes the parts lists are generated with pre-determined average areas. Every part generation situation (rectangular area to divide and minimum dimension) tends to produce lists of parts with a particular 'natural' average part aspect ratio. All the parts lists used have 150 parts, which will allows the list aspect ratio to settle to a value close to this natural aspect ratio. As the results from a number of different parts lists are averaged it is considered unnecessary to hold the aspect ratio of the parts to a particular value. However, by altering the dimensions of the rectangular area to be divided the average part list aspect ratio can be altered.

Fixed Area and Aspect Ratio Method

To test the effect of the size of a parts list on the nesting efficiency, a set of lists, each containing a different number of parts, were generated. The results are considered to be more significant if the lists in each set have the same average area and average aspect ratio. Although the Fixed Area Method guarantees a common average part area and average part dimension in a set of parts lists, the average part aspect ratio of each list still varies. This was overcome by setting a target aspect ratio and repeating the Fixed Area Method until a list with the required average aspect ratio was produced.

Every rectangular area to be divided will tend to produce a list of parts with a particular stable average part aspect ratio. For the purpose of testing the affect of list size, the actual value of the aspect ratio is irrelevant providing it is common to all

parts lists. Thus the dimensions of the rectangular areas to be divided were not adjusted to give a particular aspect ratio.

If an aspect ratio target is set which differs significantly from the natural aspect ratio, it may not be possible to replicate this value when generating a large list. If the difference is very great it may not be possible to replicate it even with a list containing a very small number of parts. The best approach is to create a few large lists of parts to find the approximate natural aspect ratio. This can then be used as the target value during the list creation. To test parts list size, this method was used to generate two sets of parts lists, each containing 160, 80, 40 ,20 and 10 parts. The two lists of 160 parts passed through less than five iterations before the average target aspect ratio occurred. One 10 part list passed through in excess of 50 iterations before the target value occurred.

4.6 The RECTNEST System

The RECTNEST system [code listed in Scott and Mileham (1996)] is the demonstration version of the new nesting system and has been written to be user friendly, with menu options and clear prompts for information to be entered. It also contains a facility to generate parts lists manually or by random generation. The nesting system or the parts list generator are selected from the main menu (Figure 4.16). Figure 4.17 shows the screen during random parts list generation. The part generation constraints are entered into a box in the top left corner of the screen. The actual parts generated are shown in a large box at the bottom of the screen and the parts list statistics are given in a box at the top right of the screen. With the manual parts list generation system, the box in the top left of the screen in Figure 4.17 is replaced by a box containing the dimensions of the last part entered.

The nesting algorithm, the layout constraints (ie. guillotine cut and part rotation) and the parts list to be placed are selected from menus (Figure 4.18). The layouts subsequently generated are displayed with all the relevant information placed in boxes around the layout drawing (Figure 4.19). When more than one sheet is used to nest a parts list they can be toggled through by hitting any key to move to the next.

Borland Turbo C contains a number of graphics sub-routines, for instance to draw a line or a rectangle. Higher level sub-routines can be constructed from these, for example to draw a menu box of a specified size and position. To represent the sheet as large as possible for clarity a fixed scale is not used. A maximum X and Y screen dimension is set for the sheet and the sheet dimensions are both scaled to meet these exactly. As the aspect ratio of the sheet and parts must be retained, the smaller of the X and Y scale factors calculated is accepted for both dimensions. The position of the sheet drawing must be off-set from the screen origin, which is the top left corner, to give them their correct position on the screen. This effectively translates the sheet drawing into the required position. This must also be done for all the menu and data boxes. No further details will be given of the system's graphics as they are not a central element of this work.

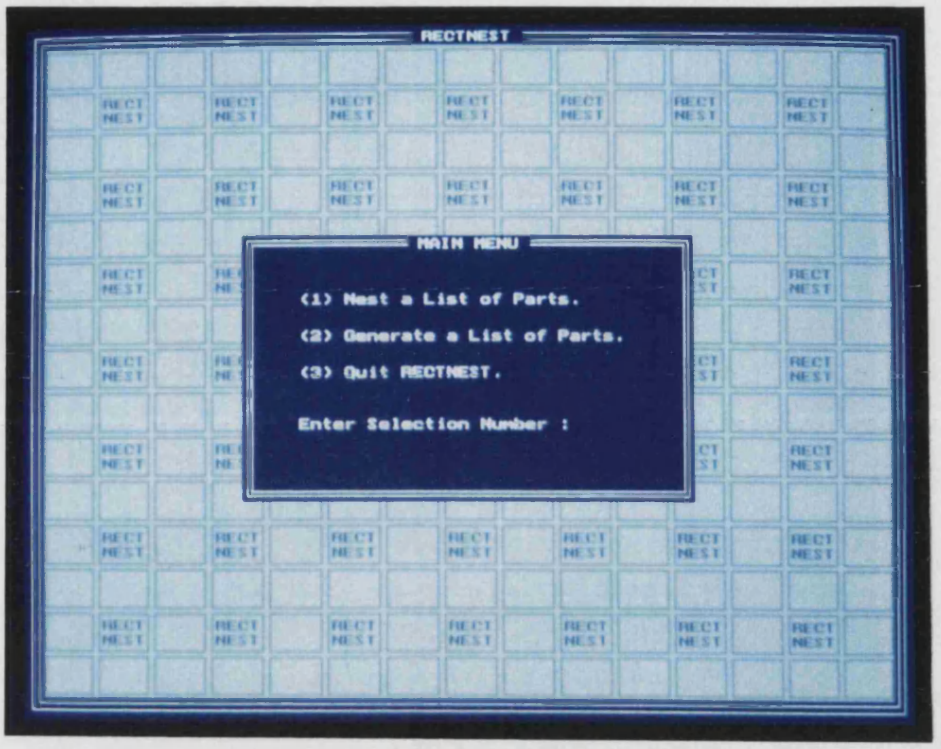


Figure 4.16.
The RECTNEST Main Menu.

Figure 4.17.
The Random Parts List Generation Screen.

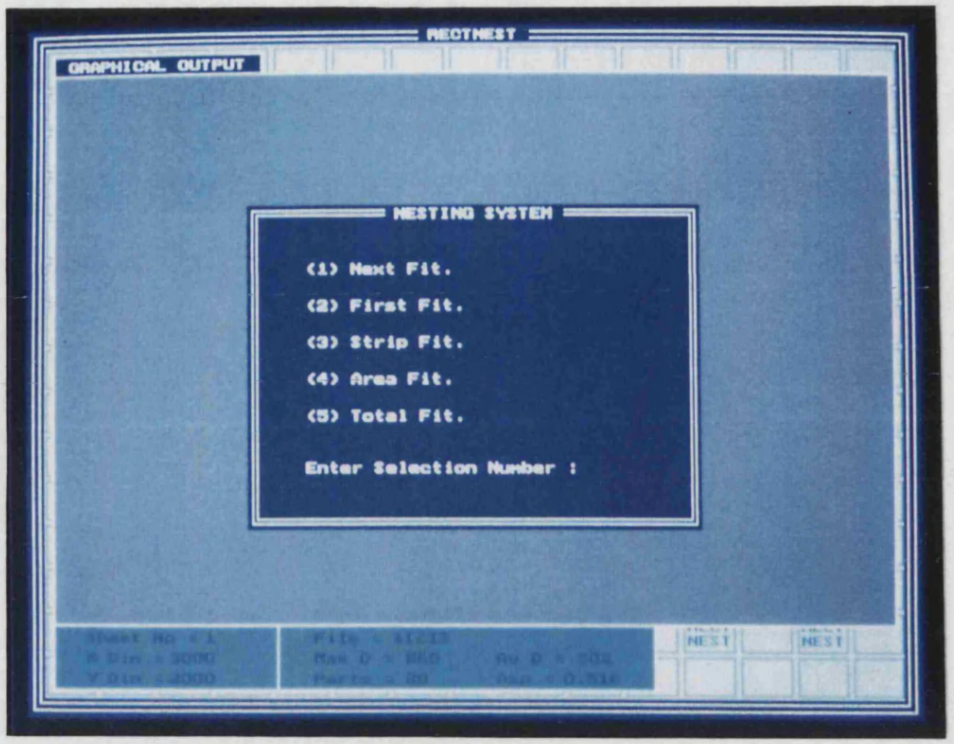


Figure 4.18.
The Nesting System Input Menus.

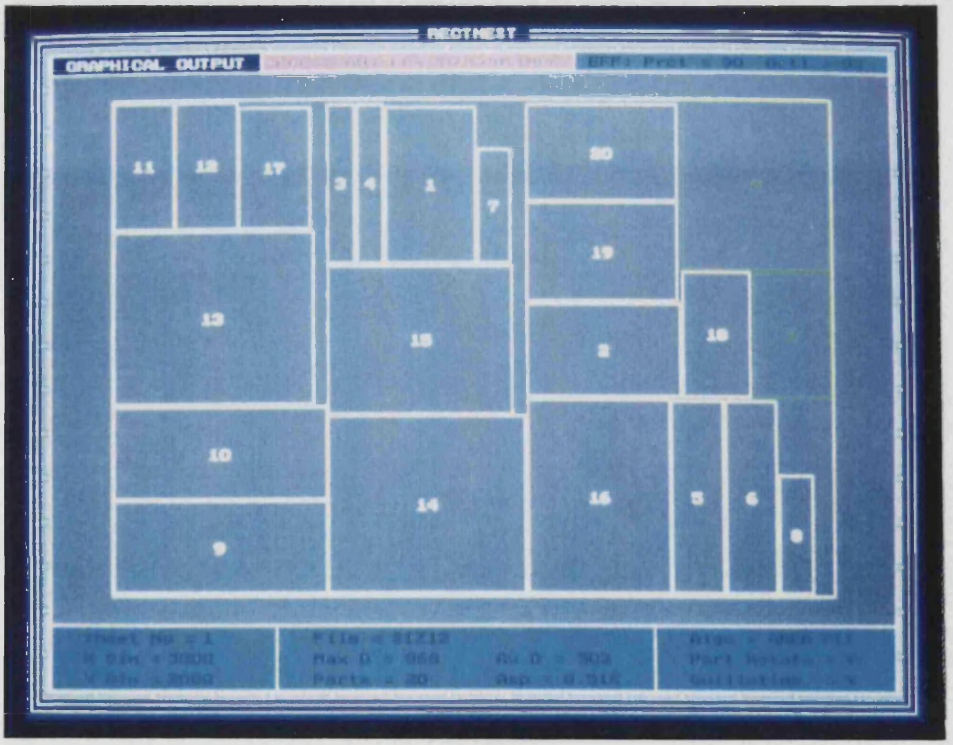


Figure 4.19.
The Sheet Layout Screen.

4.7 The TESTNEST System

Testing the algorithms using the RECTNEST system would be unnecessarily time consuming and the performance statistics would have to be noted manually. This has led to the development of the TESTNEST system. This system does not contain the graphics of the RECTNEST system or a parts list generation system as it requires parts lists to be generated by separate programs. For each parts list the TESTNEST system applies all possible algorithm and layout constraint combinations. This results in eighteen individual layouts being created for each parts list and sheet size. All of the five individual systems except the Area Fit system can have the Guillotine Cut constraint removed, giving nine variants. All of these can be applied with the parts constrained to a fixed orientation or with 90° part rotation permitted.

The system outputs a file of the format shown in Figure 4.20. This contains both the actual and practical nesting efficiencies (Section 4.4) and the average of these two values, as this combined value will be used to assess the performance of the algorithms. In addition, the parts list statistics, the dimensions of the sheet and the number of sheets used by each algorithm are given. A second file which contains only the combined performance value of each algorithm is also created. This file can be directly imported into the Quattro Pro™ spreadsheet package used to carry out more detailed analysis of the results. Scott and Mileham (1996) contains the TESTNEST system and the Quattro Pro™ spreadsheets on disk.

While the TESTNEST system is operating every sheet in every layout is drawn on the screen at half second intervals to allow the operator to validify the layout. The system counts the parts in and out of each nesting algorithm for the same purpose and the basic performance statistics are printed on the screen at the end. The results analyzed comprise of 189 parts list and sheet size combinations, many of which were multiple sheet layouts, for each algorithm variant. All parts in the test layouts were correctly placed according to the rules of the algorithms.

Parts List File = SHT03

Sheet X Dim. = 1500 Sheet Y Dim. = 960

150 Parts in List. Av. Aspect Ratio = 0.452
Max. Dim = 600 Min. Dim = 30 Av. Dim = 253

Algorithm	Rot	Guil	Sht.	Actual	Pract.	Comb.
NEXT FIT	N	Y	9.0	75.6 %	75.1 %	75.4 %
NEXT FIT	N	N	9.0	75.8 %	75.1 %	75.5 %
NEXT FIT	Y	Y	8.6	79.4 %	79.0 %	79.2 %
NEXT FIT	Y	N	8.5	79.6 %	79.4 %	79.5 %
FIRST FIT	N	Y	8.2	82.4 %	82.4 %	82.4 %
FIRST FIT	N	N	8.2	82.5 %	82.5 %	82.5 %
FIRST FIT	Y	Y	8.3	81.6 %	81.5 %	81.6 %
FIRST FIT	Y	N	8.3	81.7 %	81.7 %	81.7 %
STRIP FIT	N	Y	8.4	84.2 %	80.4 %	82.3 %
STRIP FIT	N	N	8.4	83.3 %	80.4 %	81.9 %
STRIP FIT	Y	Y	7.7	92.4 %	88.3 %	90.3 %
STRIP FIT	Y	N	7.5	92.5 %	89.7 %	91.1 %
AREA FIT	N	Y	7.2	95.1 %	94.1 %	94.6 %
AREA FIT	Y	Y	7.2	96.3 %	94.3 %	95.3 %
TOTAL FIT	N	Y	7.7	88.1 %	87.9 %	88.0 %
TOTAL FIT	N	N	7.7	88.6 %	88.3 %	88.5 %
TOTAL FIT	Y	Y	7.6	91.2 %	88.9 %	90.1 %
TOTAL FIT	Y	N	7.4	93.7 %	91.2 %	92.5 %

Figure 4.20.
The performance statistics file created by the TESTNEST system.
(sht = Number of sheets used to 1 dp)

4.8 De-bugging the code

The Borland Turbo C editor contains a number of on-line debugging tools. The most common and useful of these are 'breakpoints' which halt the program's execution at a pre-determined point, the practice of 'stepping' through the program operation by operation and the use of 'watches' which report the current values of a chosen variable. However, it is often impractical to de-bug large programs which contain linked lists with only the on-line de-bugging tools due to the nature of some of the errors which arise. In the development of this system it was necessary to write a number of dedicated de-bugging programs to provide additional specific information.

If the pointer to a data structure is removed without the structure's allocated memory space being returned to free status, the structure's memory space is lost until the computer is re-booted. If this occurs frequently the available RAM may drop below a level which will support the program's execution. However, the Borland Turbo C library routines will crash if any disparity is detected between the memory occupied by the program and the quantity of memory currently allocated. Thus a sub-routine has been written which checks for the memory disparity and if any is found the program's operation is halted and the current pointer states and sub-routine in use is reported. If this routine is positioned in the sub-routines being developed the code responsible for the memory corruption can be identified and the error rectified.

Another problem encountered is the examination of the data held in the linked lists throughout the program's operation. For this purpose sub-routines have been written to temporarily halt the program and display data within the linked lists on the screen. A commonly used routine of this type halted the program after each part placement and listed on the screen the identification numbers of all parts placed and those awaiting placement along with the method of part placement. This ensured any incorrect positioning of a part in the linked lists would be identified. Some errors detected by these methods would occur only once in a dozen program runs. Each run would contain at least a few thousand operations, which makes stepping through the program impractical if there is no information on where to place a 'breakpoint'.

Chapter 5

Example Sheet Layouts

This Chapter demonstrates the operation of each algorithm in the system developed through a series of example layouts. A quantitative comparison of each algorithms' performance is carried out in Chapter 6.

5.1 Generation of Layouts

The layouts shown in this chapter (Figures 5.1 to 5.21) have been screen dumped from the nesting system developed in this research. The layout display of the demonstration version of the nesting system, 'RECTNEST', draws the nested sheet at a reduced size to allow additional information to be simultaneously displayed. As this would give a poor print of the sheet layout another version of the system, 'VIEWNEST', has been used; which only differs from 'RECTNEST' in the graphical display which it generates. In 'VIEWNEST' each sheet is drawn as large as possible on the screen, with no other information given, to allow the clearest possible print to be obtained. All lines have been given double thickness to improve print quality. All the parts on the sheet are numbered and the two largest remaining areas on the sheet are shown as a shaded block and marked with an 'A'.

To demonstrate the operation of each algorithm and provide a comparison between the layouts, one parts list has been placed onto a sheet by each algorithm and constraint variant. A list of 40 parts has been created which highlights many of each algorithms nesting features within one sheet of 4000mm by 3000mm in size. The parts in this list have an average dimension (both X and Y) of 367mm, a maximum dimension of

996mm and an average part aspect ratio of 1:2.02. This parts list and sheet combination is representative of those layouts created during the statistical testing in Chapter 6. In addition to these 'comparison' layouts three other layouts (Figures 5.5, 5.10 and 5.11) are also shown to illustrate additional features of layouts created by the Next Fit and First Fit algorithms. These use the same parts list, but are placed on smaller sized sheets. The 'Actual' and 'Practical' layout efficiencies (Section 4.4) for all the 'comparison layouts' are shown in Table 5.1. It should be noted that any efficiency value quoted in Chapter 6 is the average of these two values.

ALGORITHM	ROT	GUIL	ACTUAL EFFICIENCY	PRACTICAL EFFICIENCY
NEXT FIT	N	Y	76.5%	73.1%
NEXT FIT	N	N	88.4%	85.4%
NEXT FIT	Y	Y	81.3%	77.6%
NEXT FIT	Y	N	90.5%	87.3%
FIRST FIT	N	Y	77.6%	76.9%
FIRST FIT	N	N	89.6%	88.7%
FIRST FIT	Y	Y	83.0%	78.3%
FIRST FIT	Y	N	92.2%	88.2%
STRIP FIT	N	Y	78.4%	62.9%
STRIP FIT	N	N	86.2%	67.3%
STRIP FIT	Y	Y	84.2%	65.9%
STRIP FIT	Y	N	89.2%	68.9%
AREA FIT	N	Y&N	91.2%	86.0%
AREA FIT	Y	Y&N	92.3%	91.1%
TOTAL FIT	N	Y	88.8%	87.6%
TOTAL FIT	N	N	88.8%	87.6%
TOTAL FIT	Y	Y	87.8%	80.2%
TOTAL FIT	Y	N	92.7%	81.9%

Table 5.1. Actual and Practical efficiency values for the 'comparison layouts'.

5.2 Next Fit Layouts

The 'comparison' layouts created by the Next Fit algorithm under the four different layout constraints are shown in Figures 5.1 to 5.4. In Figure 5.1 the parts have been ordered by decreasing X dimension, in their original orientations, and placed in vertical strips from the bottom left corner of the sheet. Waste is incurred to the right of all but the first part in each strip. This waste is considerably reduced by the Strip Squeeze algorithm, which has been applied to the layout in Figure 5.2. The Strip Squeeze algorithm can only be applied to layouts when the guillotine cut constraint is not imposed. The greater the difference in the X dimensions of the member parts of a pair of strips, the more waste that can be recovered by squeezing the pair of strips together. As the first pair of strips contain a greater disparity in the X dimensions of their member parts than the second pair of strips their 'squeezing' gives a corresponding greater improvement in layout efficiency.

Figures 5.3 and 5.4 show the two layouts with part rotation permitted. All the parts have been re-orientated with their largest dimension in the X axis prior to being placed. These layouts are comprised of fewer, but wider strips. This generally incurs less waste at the ends of the strips due to the smaller resulting Y dimensions being more likely to fit small areas at the ends of the sheets. This re-orientation of the parts does not necessarily increase or decrease the waste to the right of the parts in each strip. The 'Practical Efficiency' (Section 4.4) only considers the largest remaining rectangular area. The re-orientation of the parts results in the area at the end of the last strip being wider which will increase its area. This can be a disadvantage as this area is subsequently ignored in the 'Practical Efficiency' calculation.

Figure 5.5 shows the first of the two smaller 2400mm by 1800mm sheets required by the Next Fit algorithm to nest the parts list. The large free area at the end of the sheet is due to the X dimension of the next part being too large, requiring a new sheet to be started, even though parts exist in the list which could be accommodated in this area. This is the disadvantage of a pre-determined sequence of part placement. Some waste of this type is incurred at all but the last sheet nested by this algorithm.

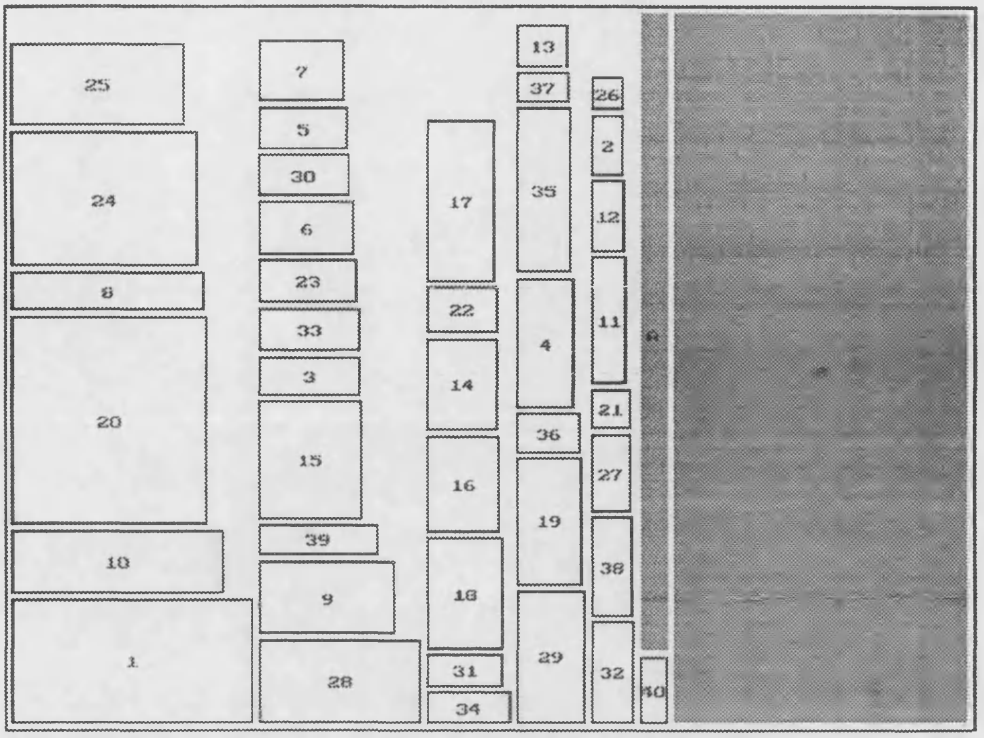


Figure 5.1.
Comparison layout - Next Fit, no Strip Squeeze, no part rotation.

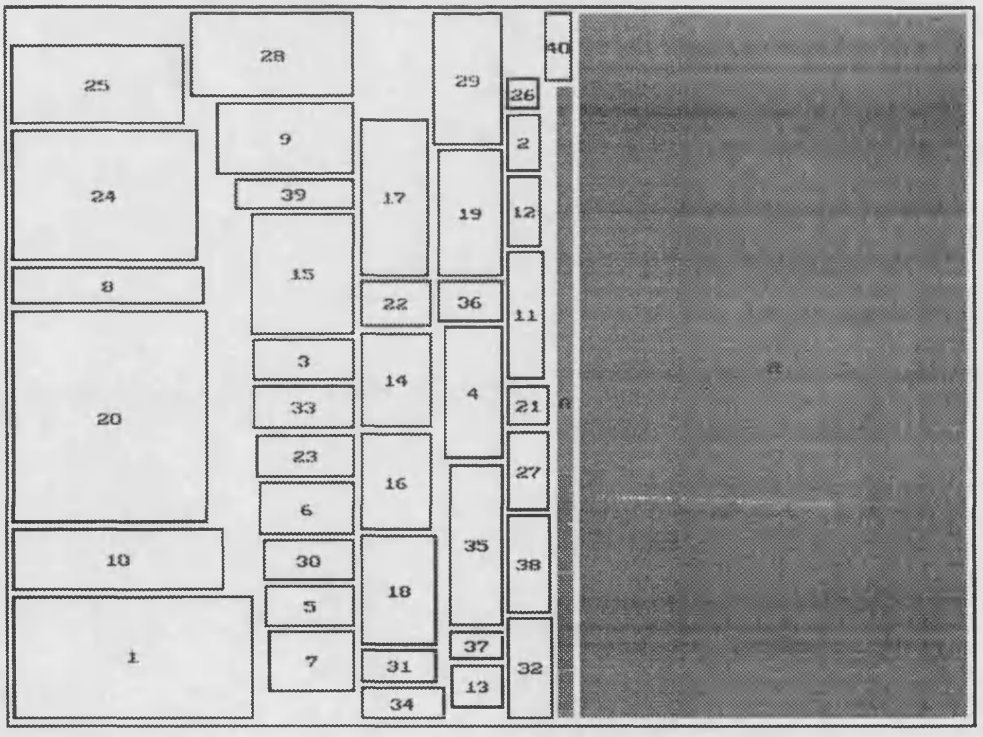


Figure 5.2.
Comparison Layout - Next Fit, Strip Squeeze applied, no part rotation.

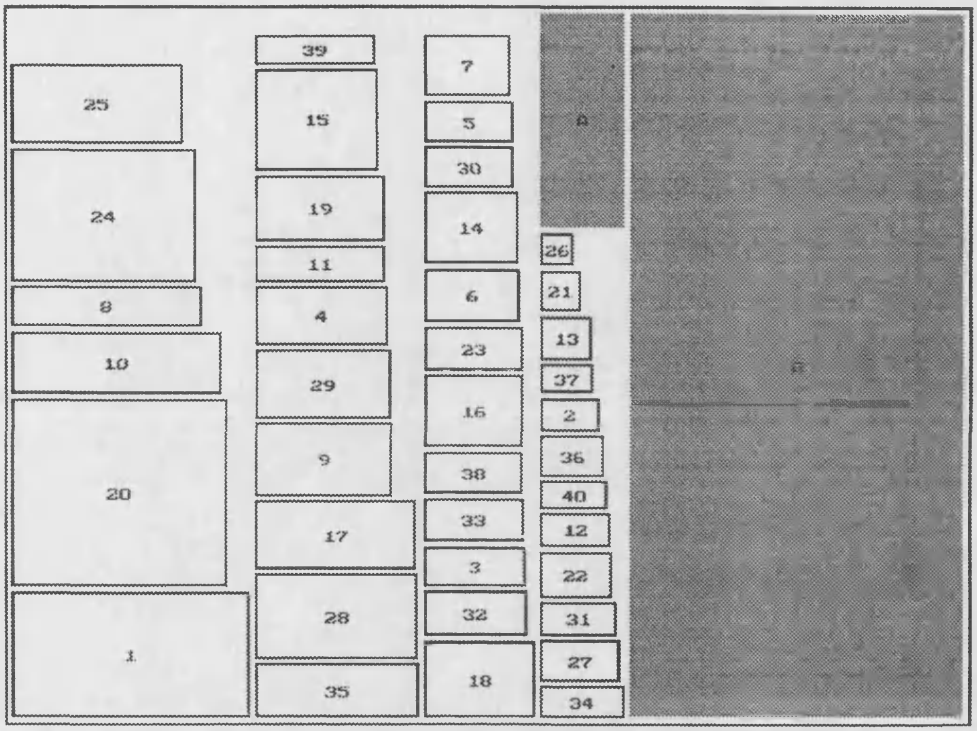


Figure 5.3.
Comparison Layout - Next Fit, no Strip Squeeze, part rotation permitted.

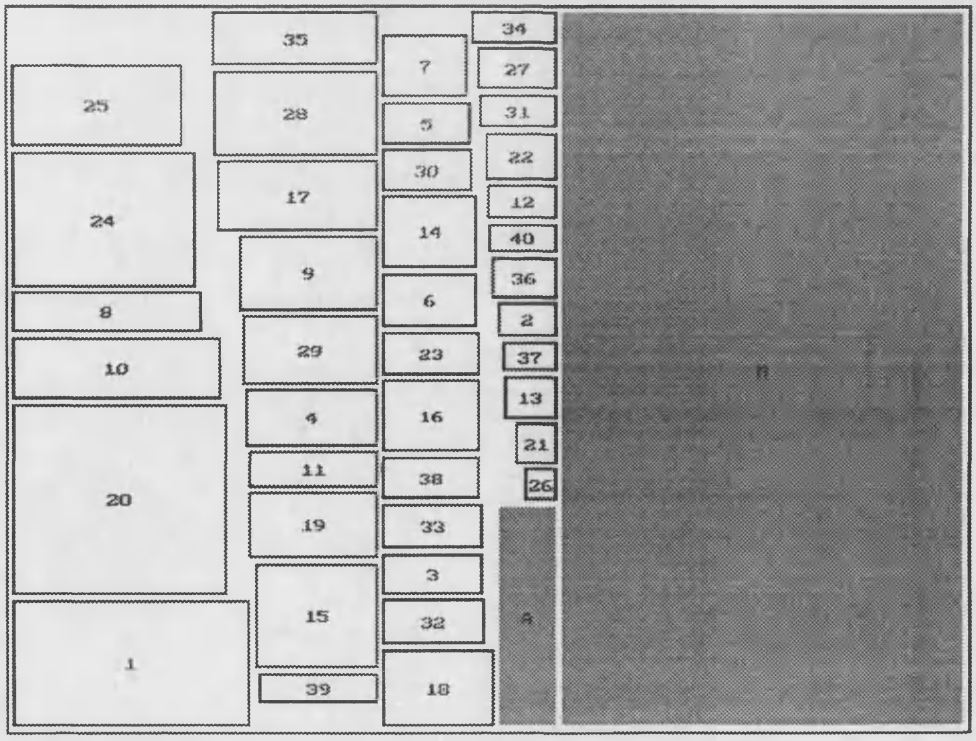


Figure 5.4.
Comparison Layout - Next Fit, Strip Squeeze applied, part rotation permitted.

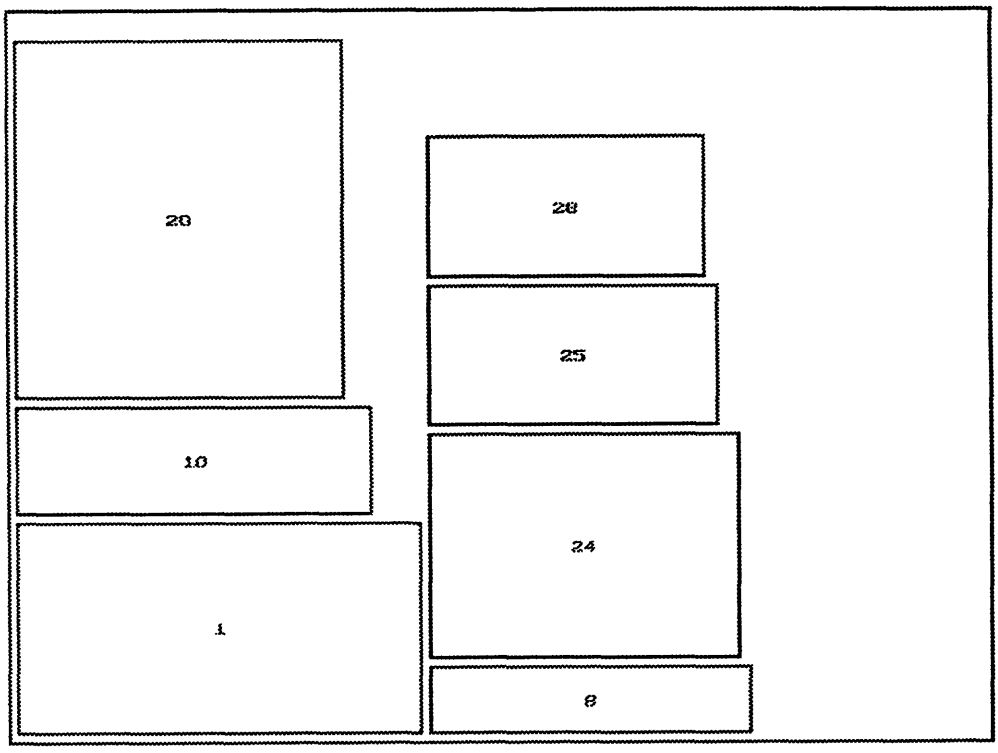


Figure 5.5.
End of sheet waste incurred with the first of a multiple sheet Next Fit algorithm (ROT N & GUIL Y) layout.

5.3 First Fit Layouts

The 'comparison' layouts created by the First Fit algorithm under the four different layout constraints are shown in Figures 5.6 to 5.9. This algorithm is an adaption of the Next Fit algorithm. Parts are placed in a pre-determined sequence, however the areas at the end of every formed strip are held and parts are placed into these areas in preference to their conventional location. In Figure 5.6 the end areas of the first two strips are too small to accommodate any of the remaining parts in the list. The first difference between this layout and the equivalent Next Fit layout (Figure 5.1) occurs in the third strip where two additional parts, 36 and 37, are placed in the strip's 'end area'. In this particular case the larger of the layout's two remaining areas is considerably increased, giving a significant improvement in the 'Practical Efficiency'. Removing the guillotine cut constraint (Figure 5.7), allowing part rotation (Figure 5.8), or removing both layouts constraints (Figure 5.9) has the same effect on the layouts of the First Fit algorithm as on those of the Next Fit algorithm.

Both the Next Fit and First Fit algorithms required three of the smaller 2000mm by 1500mm sheets to accommodate the example parts list (with the guillotine cut constraint imposed and part rotation not permitted). Figure 5.10 shows the first sheet layout created by the Next Fit algorithm and Figure 5.11 shows the first sheet layout created by the First Fit algorithm. This sheet size has been chosen because it gives an extreme example of the layout improvement which can be achieved by placing parts into strip 'end areas' ie. the benefit of using the First Fit algorithm rather than Next Fit algorithm. In this particular situation three additional parts have been placed on the sheet which vastly improves the layout efficiency. Generally, the smaller the Y dimension of the sheet, the greater the disparity between the layouts of the First Fit algorithm and those of the Next Fit algorithm.

In the previous section Figure 5.5 illustrated the waste incurred at the end of a sheet due to the next part being too large for the remaining area and a new sheet being started. The First Fit algorithm could be adapted to fill this area by treating it as an end area of a strip.

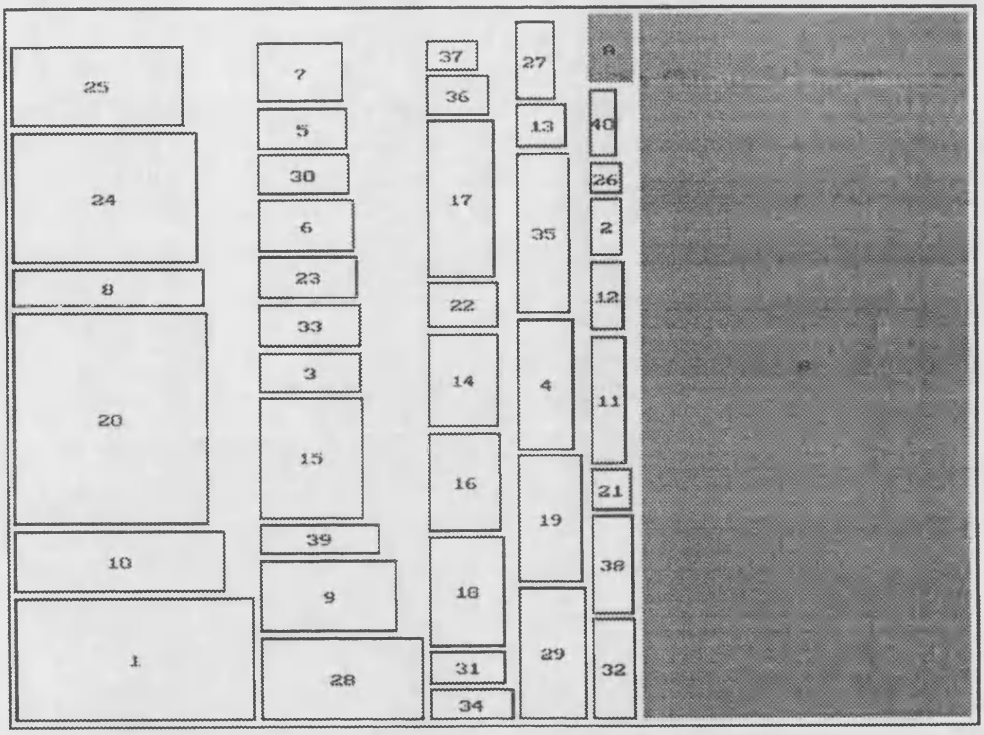


Figure 5.6.
Comparison layout - First Fit, no Strip Squeeze, no part rotation.

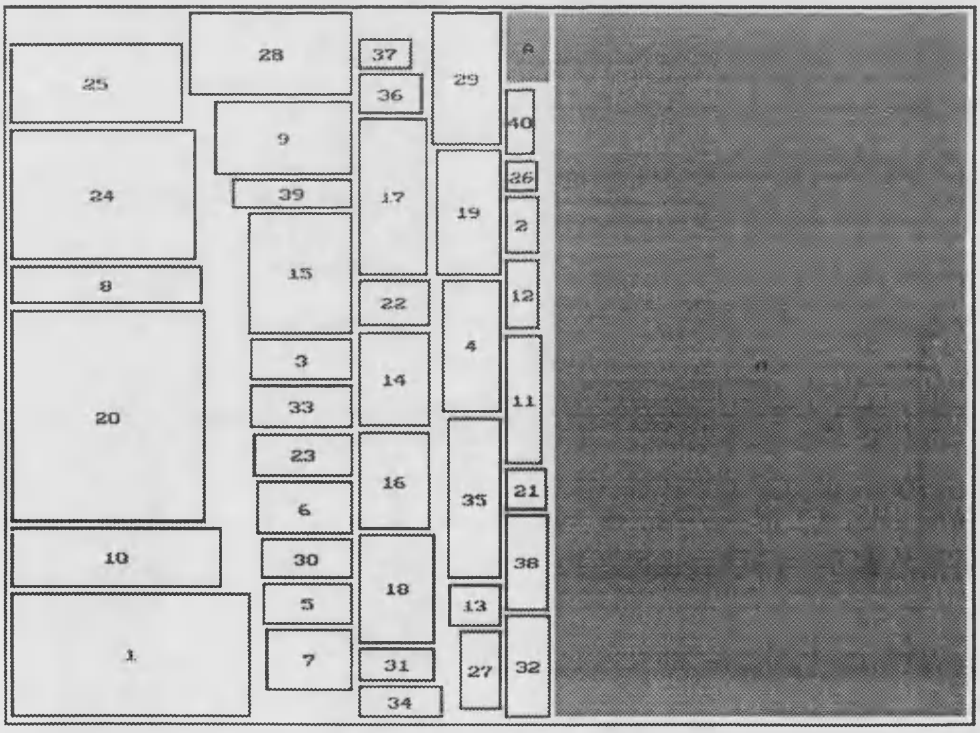


Figure 5.7.
Comparison Layout - First Fit, Strip Squeeze applied, no part rotation.

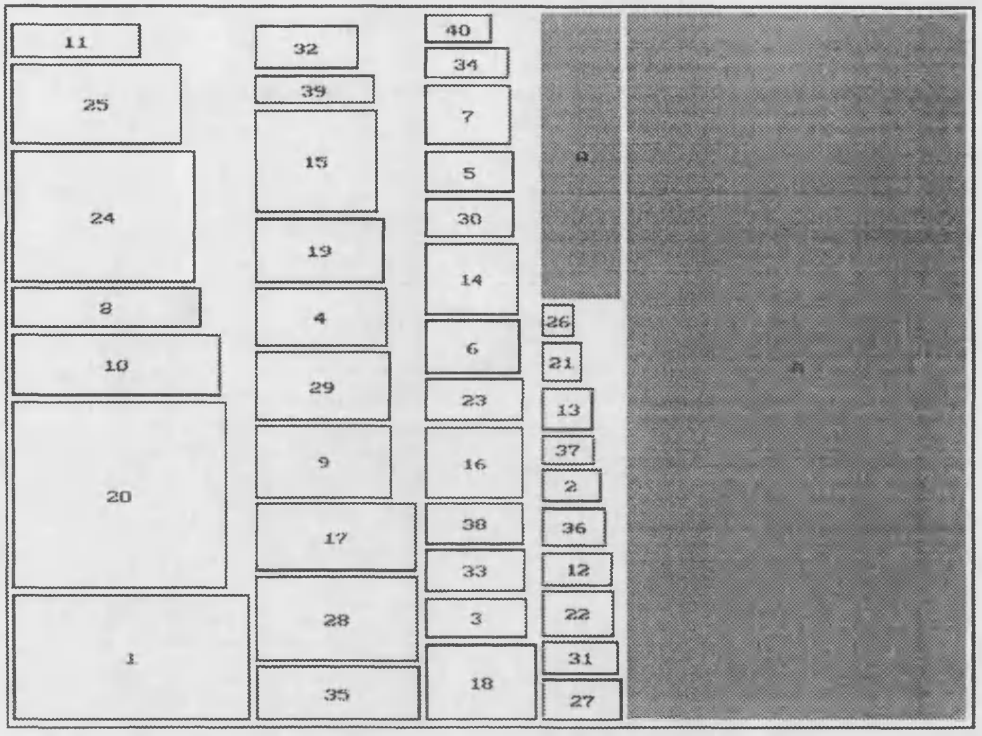


Figure 5.8.
Comparison Layout - First Fit, no Strip Squeeze, part rotation permitted.

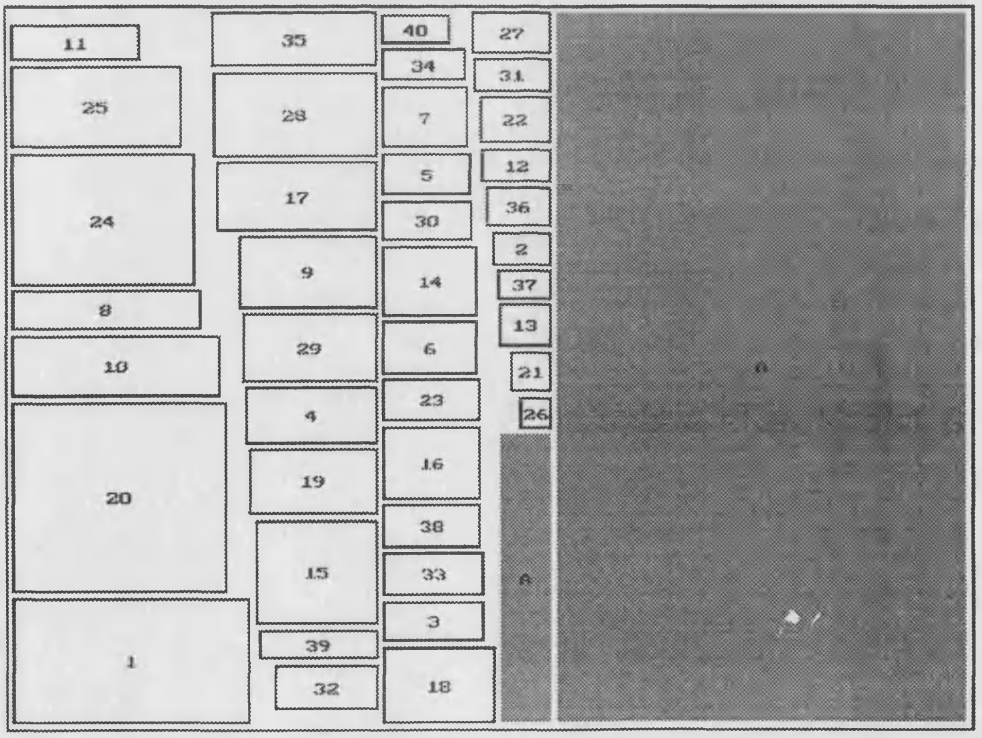


Figure 5.9.
Comparison Layout - First Fit, Strip Squeeze applied, part rotation permitted.

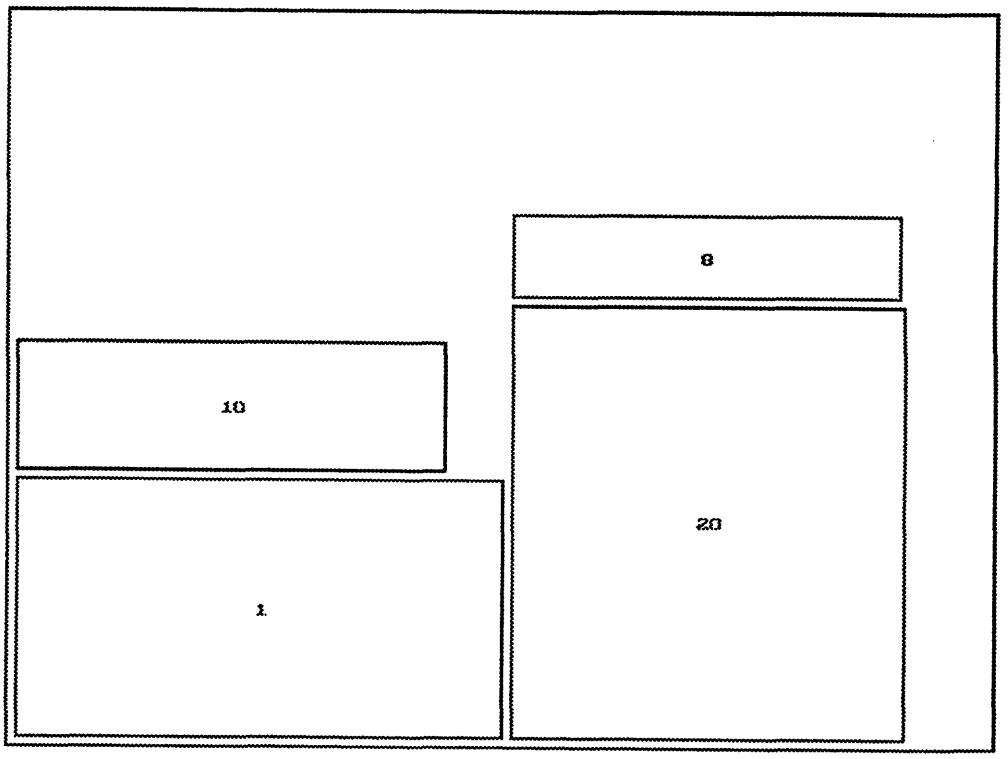


Figure 5.10.
Next Fit layout on a comparatively small sheet - no end area placement.

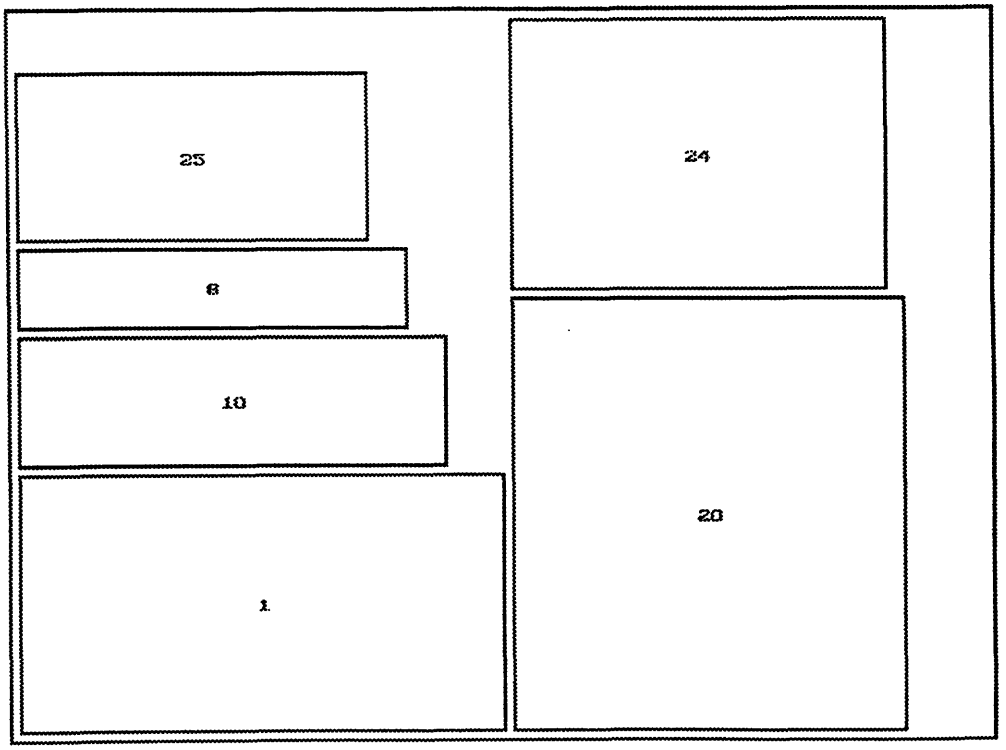


Figure 5.11.
First Fit layout on a comparatively small sheet - with end area placement.

5.4 Strip Fit Layouts

The 'comparison' layouts created by the Strip Fit algorithm under the four different layout constraints are shown in Figures 5.12 to 5.15. The Strip Fit algorithm (Section 3.6) gathers parts within a width tolerance and attempts to form a strip of these parts with a length which closely fits the sheet's Y dimension. The upper bound of the width tolerance is progressively moved from the largest part dimension in the list to the smallest.

This algorithm is intended to only nest some of the parts list and then hand over to another algorithm. To completely nest a parts list by this method it is necessary to repeatedly relax the acceptance tolerances until the last part has been placed. Figure 5.12 shows the 'comparison layout' with both layout constraints imposed. It can be seen that from left to right the strips generally become shorter and there is an increasing disparity in the X dimensions of the member parts. To place the last part in this example, the strip length tolerance has been sufficiently relaxed to accept the part's Y dimension as a suitable strip length.

The 'squeezed' layout (guillotine cut constraint removed) is shown in Figure 5.13. The strips to the right of the layout have a more tapered form offering greater potential for improvement. The two layouts with part rotation permitted are shown in Figures 5.14 and 5.15. It is clear from these layouts that part rotation allows far more efficient strips to be formed; very little strip tolerance relaxation has taken place before the creation of the sixth strip.

The least constrained layout, Figure 5.15, highlights why many researchers advocate intervention by a human planner to improve a layout. Moving part 26 to the bottom of the second row and translating part 20 to the left would give a considerable material saving.

One major disadvantage with this algorithm is that it often places large parts last, dividing the remaining sheet into two large rectangular areas. One of these areas has

to be discarded in the calculation of the Practical Efficiency (Section 4.4). The effect of this can be seen in the difference between the Practical and Actual efficiencies for this algorithm (Table 5.1, Section 5.1). In all four Strip Fit layouts shown a larger single rectangular area could be created by taking a horizontal rather than vertical dividing line from the top right corner of the last part. This is not permitted for the purposes of waste evaluation as it is not the division of the sheet which the algorithm makes.

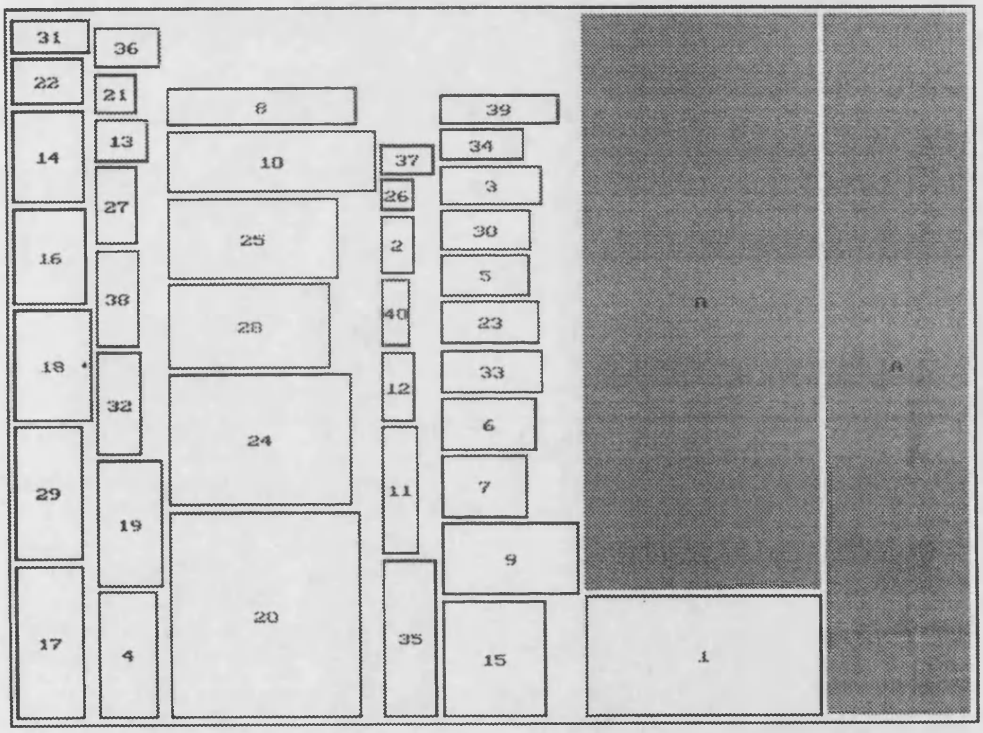


Figure 5.12.
Comparison layout - Strip Fit, no Strip Squeeze, no part rotation.

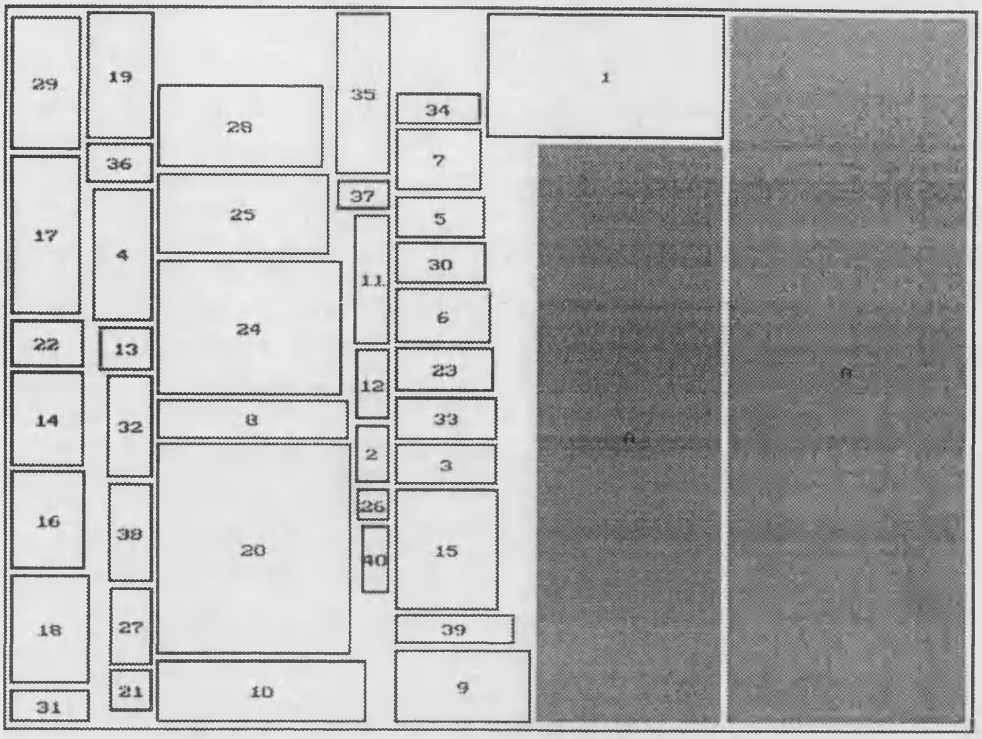


Figure 5.13.
Comparison Layout - Strip Fit, Strip Squeeze applied, no part rotation.

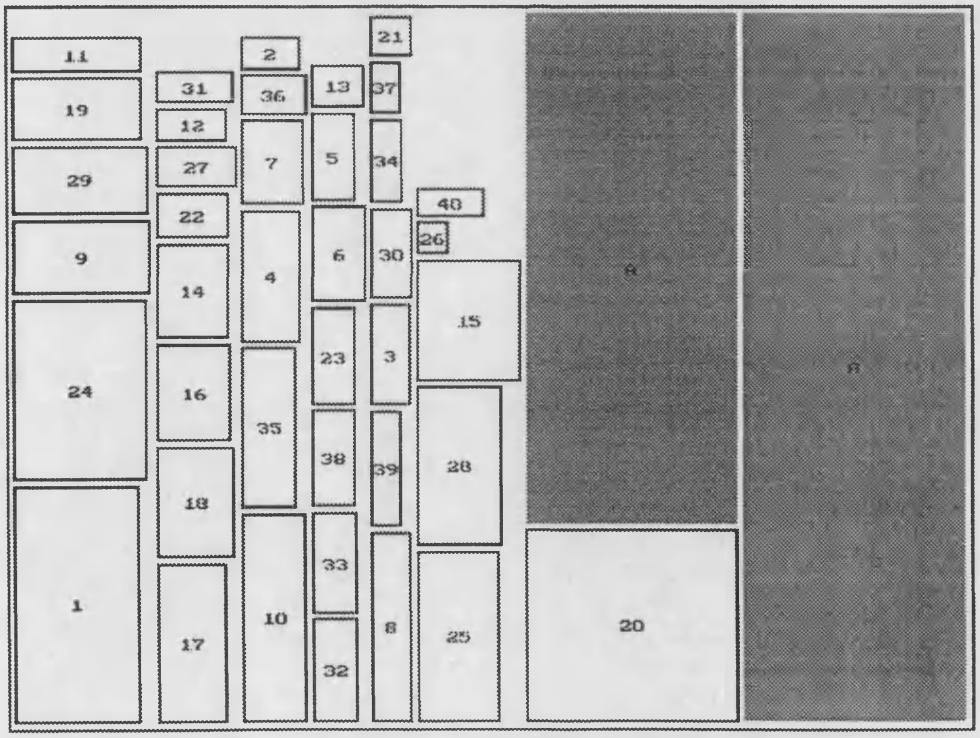


Figure 5.14.
Comparison Layout - Strip Fit, no Strip Squeeze, part rotation permitted.

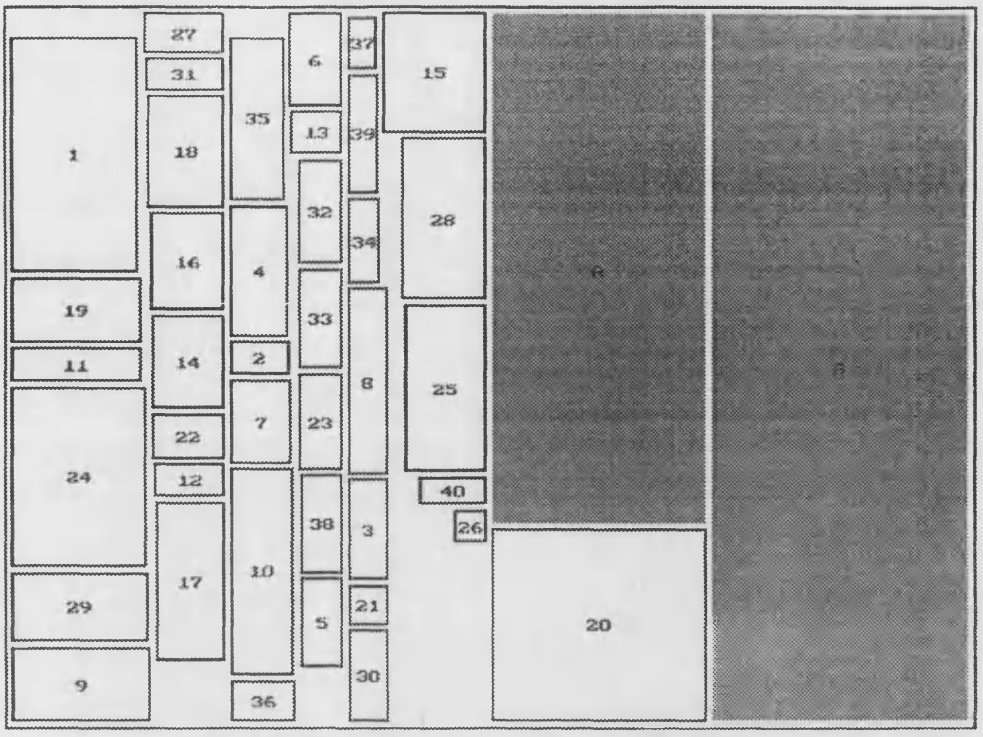


Figure 5.15.
Comparison Layout - Strip Fit, Strip Squeeze applied, part rotation permitted.

5.5 Area Fit Layouts

The two 'comparison' layouts created by the Area Fit algorithm under the four different layout constraints are shown in Figure 5.16 and 5.17. The Area Fit algorithm (Section 3.7) produces guillotine cut constrained layouts which cannot be improved by the Strip Squeeze algorithm. Thus the guillotine cut constrained layouts shown in Figures 5.16 and 5.17 would also be produced when the constraint is not imposed. The algorithm considers the current space on the sheet to nest and searches for a part in the list which can be accommodated and has an area in excess of 90% of that of the space. If no suitable part is found the algorithm searches for a part with one dimension within 90% of one of the space's dimensions. Finally, if no part has been found, the algorithm places the largest possible part in the bottom left of the space and divides the sheet around it.

Generally, permitting part rotation is of considerable advantage to this algorithm, however there are local situations where this is not the case. With part rotation (Figure 5.17) the second part placed, number 20, is orientated with its largest dimension in the X axis which does not permit the placement of parts 11 and 12 at its side, as in Figure 5.16. Overall, however, the layout with part rotation is more efficient than the layout without part rotation.

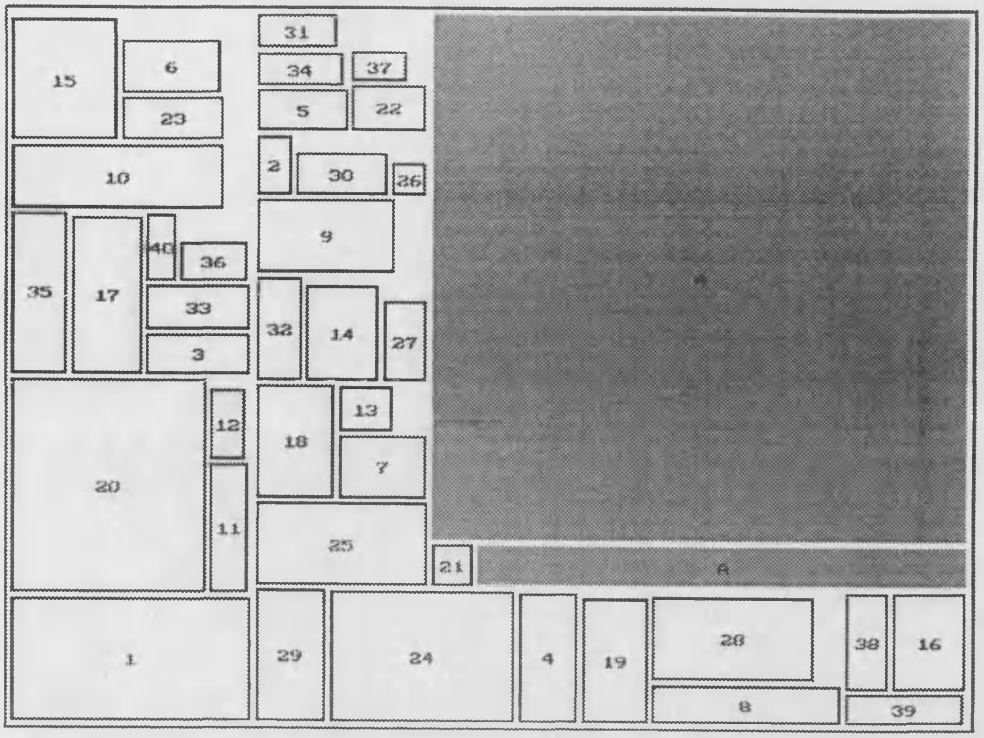


Figure 5.16.
Comparison layout - Area Fit, cannot Strip Squeeze, no part rotation.

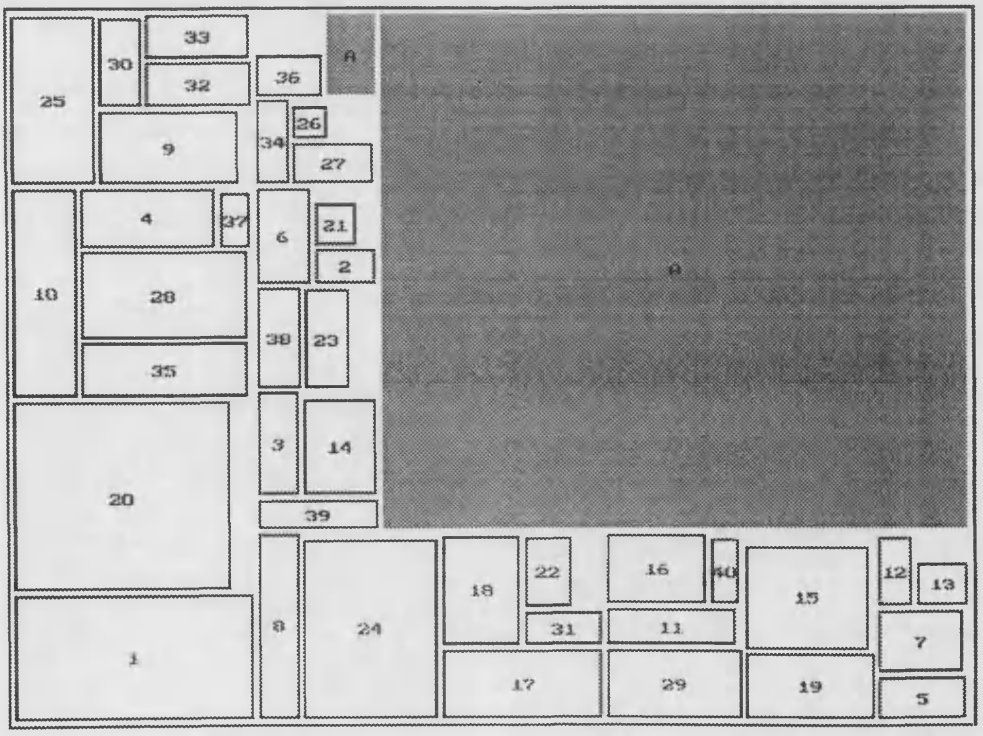


Figure 5.17.
Comparison Layout - Area Fit, cannot Strip Squeeze, part rotation permitted.

5.6 Total Fit Layouts

The 'comparison' layouts created by the Total Fit algorithm under the four different layout constraints are shown in Figures 5.18 to 5.21. The Total Fit algorithm initially applies the Strip Fit algorithm with length and width acceptance tolerances of 90%. When all the possible strips have been formed within these tolerances, rather than relaxing the tolerances, the remaining parts are handed to the Area Fit algorithm for placement. Within the Total Fit algorithm the Strip Fit algorithm is used in the manner in which it was originally intended to operate.

Figure 5.18 shows the 'comparison layout' with both layout constraints imposed. Only one strip has been formed by the Strip Fit algorithm before handing over to the Area Fit algorithm. If a small parts list is to be placed on a sheet with a large Y dimension the Strip Fit algorithm may not be able to form any strips and the entire layout will be formed by the Area Fit algorithm. When the guillotine cut constraint is removed, Figure 5.19, the layout does not change as it does not contain two strips to squeeze together.

The two layouts with part rotation permitted are shown in Figures 5.20 and 5.21. With part rotation permitted the Strip Fit algorithm is able to form four strips before handing over the nesting to the Area fit algorithm. More than two strips exist in this layout and therefore the relaxation of the guillotine cut constraint is of benefit, as the Strip Squeeze algorithm can make an improvement to the layout. The resulting slight increase in the X dimension of the area to be nested by the Area Fit algorithm changes the layout produced, with parts 6 and 26 adopting new positions.

On average, allowing part rotation gives an improvement in layout efficiency, however this is not guaranteed for every case. In the unconstrained 'comparison layout' (Figure 5.21) the 'Actual' efficiency is the highest of all the Total Fit layouts, however the 'Practical' Efficiency is not the highest. This is due to the remaining area being more evenly spread between the two remaining spaces (marked 'A' in the figure), the smaller of which is ignored in the calculation of the 'Practical Efficiency'.

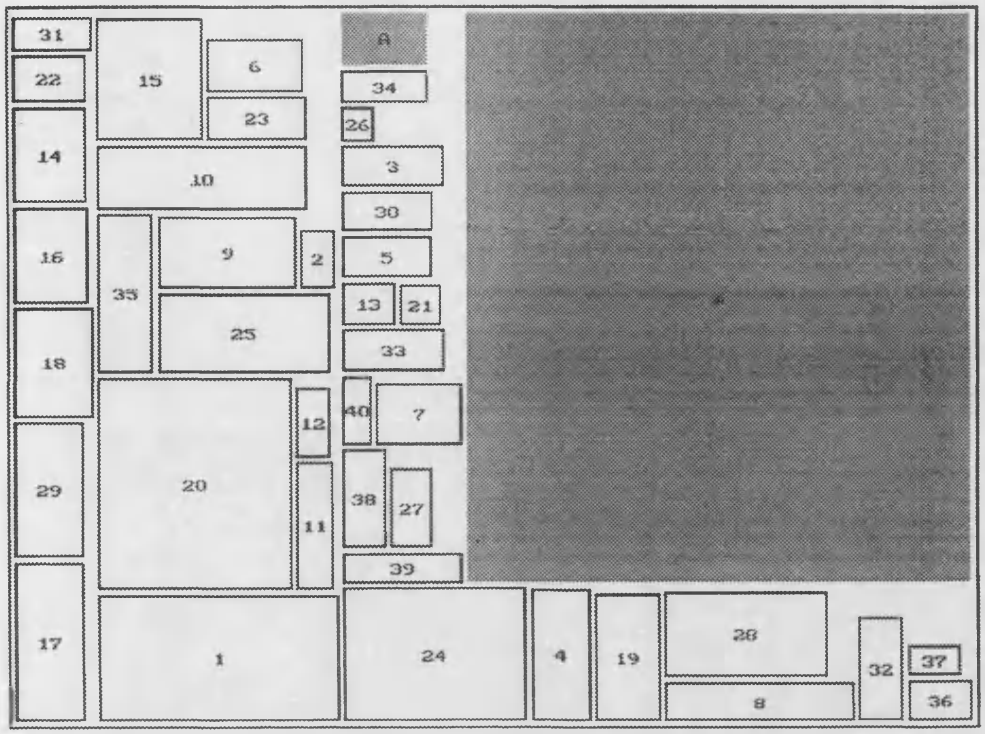


Figure 5.18.
Comparison layout - Total Fit, no Strip Squeeze, no part rotation.

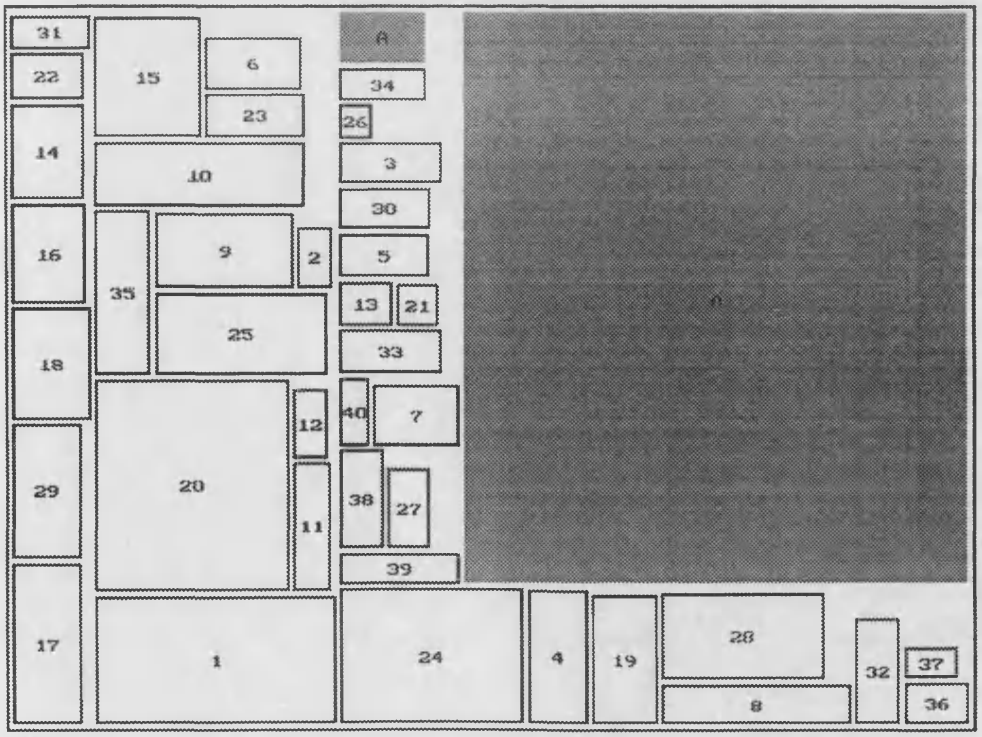


Figure 5.19.
Comparison Layout - Total Fit, Strip Squeeze permitted, no part rotation.

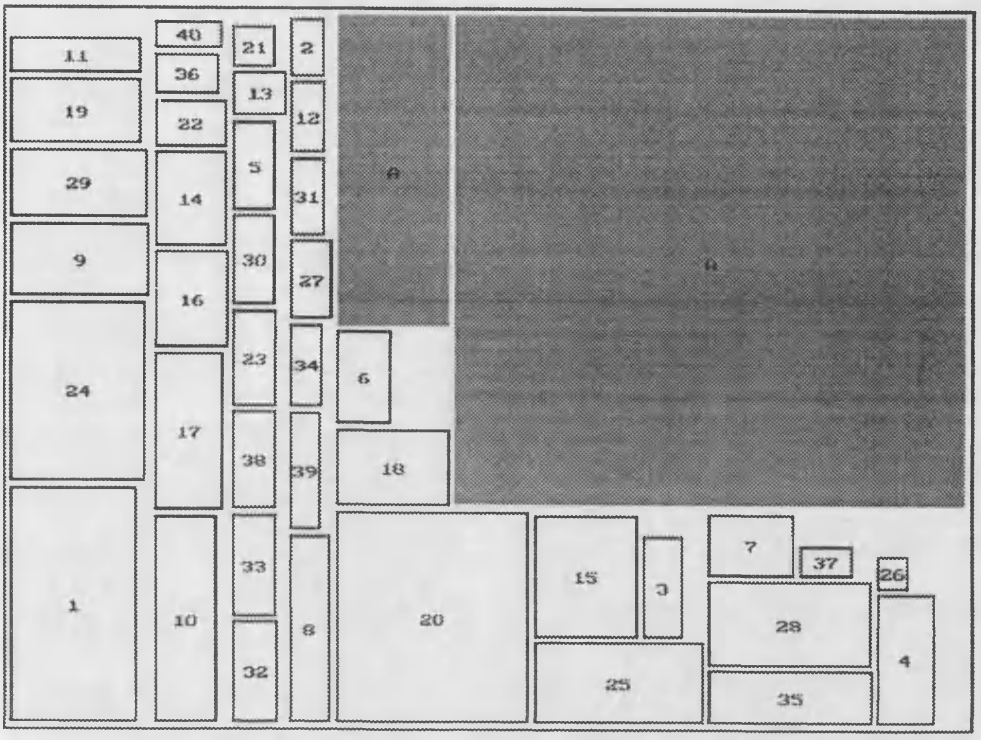


Figure 5.20.
Comparison Layout - Total Fit, no Strip Squeeze, part rotation permitted.

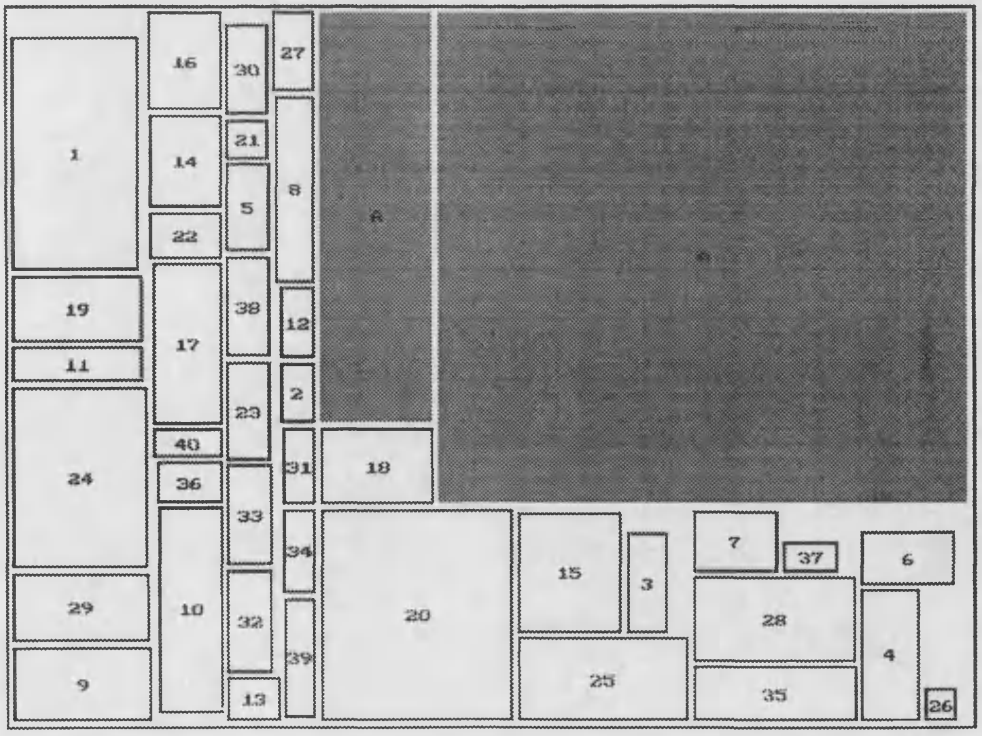


Figure 5.21.
Comparison Layout - Total Fit, Strip Squeeze applied, no part rotation.

Chapter 6

Testing the Nesting Algorithms

This Chapter contains the results of detailed quantitative testing of the nesting system developed. Section 6.2 compares the overall performance of each constraint variant for each algorithm. Sections 6.3 to 6.6 examine the effects of four nesting problem parameters on the nesting efficiencies of the algorithms.

6.1 Introduction to Testing

All the tests described in this chapter were carried out using the TESTNEST system described in Section 4.7. Analysis of each algorithm's performance in terms of both the 'Actual' and the 'Practical' layout efficiency (defined in Section 4.4) would be excessively laborious. Therefore, all performance statistics quoted are the average of the layout's 'Practical' and 'Actual' efficiency values.

The 'Actual' efficiency treats all remaining spaces on the sheet, which the algorithm could potentially continue to fill with parts, as reusable. This takes no account of the area or dimensions of these spaces, which may be too small to accommodate parts. Thus this measure does not consider the practicalities of reusing the remaining spaces on the sheet. The 'Practical' efficiency recognises these practical limitations by only considering the largest remaining space on the sheet, providing it exceeds a size limit. However, this approach is considered to be excessively strict to be used alone, as it will often 'dampen' the disparities between the performances of the algorithms. An example is if two algorithms have created layouts which both require the same number

of sheets, however one fully occupies the last sheet where as the other leaves some space. If this remaining space is too small to qualify for inclusion in the calculation of the 'Practical' efficiency, the layouts of both algorithms will be given the same efficiency value, although one layout is clearly better. Thus the use of the 'Practical Efficiency alone also has disadvantages. By using the average of the 'Actual' and the 'Practical' efficiency values, equal weighting is given to the benefits of each method.

Five placement algorithms were tested in all. The Next Fit and First Fit [Coffman et al (1980)] were used as benchmarks. The Area Fit and Strip Fit algorithms were used both individually and in combination as the Total Fit algorithm. The Strip Squeeze algorithm was used to improve the layouts of all but the Area Fit algorithm when the guillotine cut constraint was not imposed.

Performance depends on the parts list size, the aspect ratio of the parts, the aspect ratio of the sheet and the size of the sheet in relation to the parts. Each algorithm may deal with some or all of these parameters better than the other algorithms. Thus a set of tests have been devised which comprehensively test all of these parameters. In total 189 combinations of part list and sheet size were tested on each of the 18 different algorithm and nesting constraint variants (3402 individual algorithm runs). All of the test results were used to evaluate the general performance of the algorithms for each of the four types of layout constraint. Of the 189 tests, 48 considered the effect of the 'average part area to sheet area ratio' (ie. the size of the sheet) on the efficiency of layouts produced. Another set of 48 tests were specifically aimed at identifying the effect of average part aspect ratio on the efficiency of layout's produced and a further 45 of the tests examined the effect of nesting different parts list sizes. The remaining 48 were used to examine the effect of the sheet aspect ratio on the layout efficiency. An output file of the format shown in Figure 4.16 (Section 4.7) was generated for each combination of part list and sheet size. These, along with the test data files, are archived [Scott and Mileham (1996)].

The tests carried out cover a broad range of nesting situations allowing a comprehensive assessment of each algorithm's performance to be made. Parts list

sizes varied from 10 to 160 parts. The parts within each list varied greatly in their size (area) relative to the sheet onto which they were placed. The largest parts had, on average, approximately $^1/_{10}$th of the sheet's area and the smallest parts approximately $^1/_{400}$th of the sheet's area. A wide range of average part aspect ratios and sheet aspect ratios were also tested.

The performance histograms of all variants of the five algorithms tested had negatively skewed distributions, the values of Pearson's measure of Skewness (Sk_p) are given in Table 6.1. The formula for this measurement is given in Appendix B. This prevents the commonly used 't' and 'z' tests being applied to the full data set as they assume a normal distribution. Thus the Mann-Whitney or 'U' test, which does not assume a normal distribution of data was used to evaluate the significance of the test results. Details of the use of this test are given in Appendix B. These tests and the statistical terms used in this work are covered in more detail by Harnett and Murphy (1985), Becker and Harnett (1987), Mansfield (1987) and Nolan (1994). Also, a C program was written to carry out the Mann-Whitney test on the results.

	ROT N & GUIL Y	ROT N & GUIL N	ROT Y & GUIL Y	ROT Y & GUIL N
NEXT FIT	-0.54	-0.47	-1.01	-0.90
FIRST FIT	-0.98	-0.95	-1.48	-1.43
STRIP FIT	-1.61	-1.71	-1.60	-1.77
AREA FIT	-2.23	-2.23	-2.36	-2.36
TOTAL FIT	-2.22	-1.98	-1.85	-1.60

Table 6.1. Skew values for the algorithms under each layout constraint variant.

6.2 Overall Performance of the Algorithms

Purpose of the Test

This section examines the overall performance of the algorithms tested in terms of the layout efficiencies achieved, without examining the relationships between the particular parts list and sheet characteristics which were used. These characteristics are examined in Sections 6.3 to 6.6, where the details of the sheet dimensions and parts list features used for each layout are described in detail. The statistics quoted for each variant of each algorithm are taken from a total of 189 test combinations of parts list and sheet size.

Results of the Test

The mean layout efficiencies for each variant of each algorithm are shown in Figure 6.1. As the Area Fit algorithm cannot be improved by the Strip Squeeze algorithm, only a layout which is guillotine cut constrained can be created. Thus in Figure 6.1 the average efficiency value for constraint variant (1) is the same as that for constraint variant (2) and (3) matches (4). This is also true for the maximum (best) layout efficiency, the minimum (worst) layout efficiency and the standard deviation of this algorithm, which are shown in Figures 6.3 to 6.5. The values for Figures 6.1 and 6.3 to 6.5 are given in Table 6.2 and are based on the average of the Actual and Practical efficiency for each layout.

The Area fit algorithm has the highest mean layout efficiency of all the algorithms for all four variants, shown in Figure 6.1, but is closely followed by the Total Fit algorithm. These two algorithms were expected to create the best layouts, however as the Total Fit algorithm combines the Strip Fit algorithm and the Area Fit algorithm it was expected to give the better performance. The relatively poor performance of the Total Fit algorithm can be attributed to unexpected effects in the operation of the algorithm and in the acceptance tolerances set. It was assumed that after a set of

efficiently placed strips had been formed by the Strip Fit algorithm, the Area Fit algorithm would be able to place the remaining parts in a very efficient layout. As the parts not placed by the Strip Fit algorithm are all the awkward shapes and sizes, the resulting Area Fit layout efficiency is considerably lower than expected. Also, as the Area Fit algorithm generally creates an efficient layout on its own, the Strip Fit tolerances should, in retrospect, have been set higher. This would have led to the creation of fewer more efficient strips and would leave a larger proportion of the parts list to the Area Fit element of the system. Thus the Total Fit algorithm's performance could be improved by optimising the Strip Forming acceptance tolerances.

ALGORITHM	ROT	GUIL	MEAN	MAX	MIN	S.D.
NEXT FIT	N	Y	79.3%	91.9%	61.2%	7.2%
NEXT FIT	N	N	81.8%	94.8%	61.2%	8.1%
NEXT FIT	Y	Y	81.2%	92.5%	54.1%	7.7%
NEXT FIT	Y	N	83.2%	95.9%	54.1%	8.6%
FIRST FIT	N	Y	85.3%	94.4%	62.7%	7.0%
FIRST FIT	N	N	88.0%	97.1%	62.7%	7.1%
FIRST FIT	Y	Y	86.2%	96.0%	54.5%	6.8%
FIRST FIT	Y	N	88.4%	97.4%	54.5%	7.2%
STRIP FIT	N	Y	80.7%	94.5%	42.0%	10.1%
STRIP FIT	N	N	84.2%	95.9%	42.8%	9.8%
STRIP FIT	Y	Y	86.9%	94.9%	60.5%	6.2%
STRIP FIT	Y	N	89.5%	96.6%	61.0%	6.0%
AREA FIT	N	Y&N	90.1%	95.9%	59.1%	6.3%
AREA FIT	Y	Y&N	92.6%	97.1%	65.9%	5.4%
TOTAL FIT	N	Y	88.9%	96.3%	59.1%	5.4%
TOTAL FIT	N	N	89.7%	96.4%	59.1%	5.9%
TOTAL FIT	Y	Y	89.9%	96.3%	71.4%	4.3%
TOTAL FIT	Y	N	91.1%	97.3%	71.4%	4.9%

Table 6.2. Overall performance values for the algorithms.

The order of the remaining three algorithms in terms of mean layout efficiency is as expected. The best of the remaining three algorithms is the First Fit which, as it is an improvement of the Next Fit, will always generate layouts which are identical or more efficient layouts. The Strip Fit algorithm is also better in all the constraint situations than the Next Fit algorithm. The removal of the fixed part orientation constraint, variants 2 and 4, allows the Strip Fit algorithm to produce a better mean layout efficiency than the First Fit algorithm.

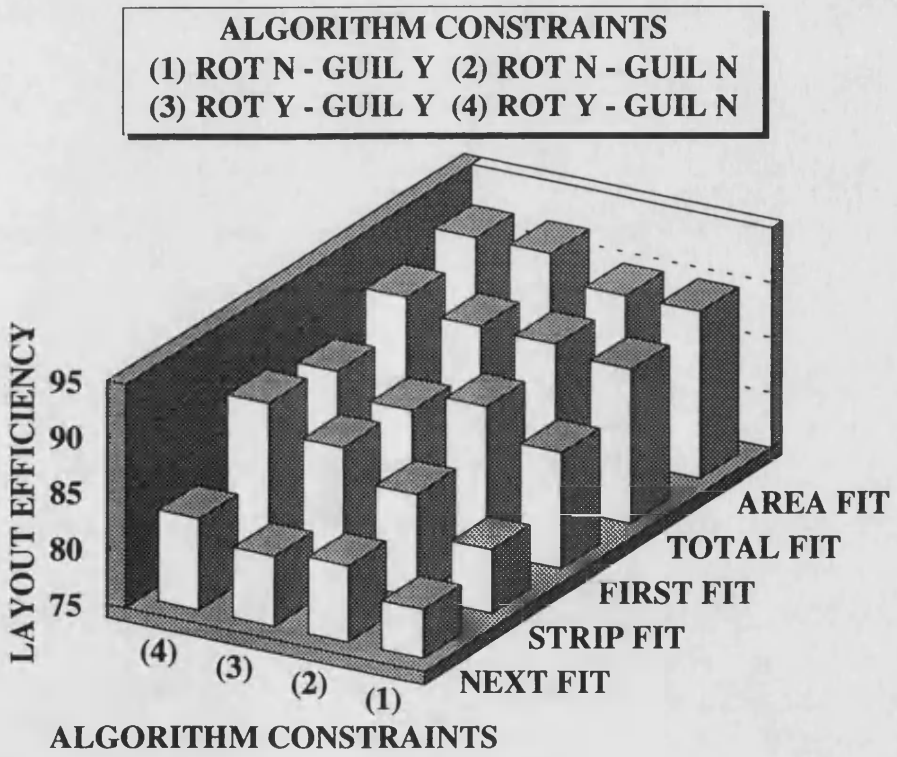


Figure 6.1.
The mean layout efficiencies for all constraint variants of each algorithm.

The affect of the constraints on mean layout efficiency achieved by each algorithm is shown more clearly in Figure 6.2, which gives the average improvements in layout efficiency for the relaxation of either or both constraints. It must be noted that, in this work, all improvements in layout efficiency associated with the removal of the guillotine cut constraint are due to the application of the Strip Squeeze algorithm. The

performance of other algorithms which can create layouts which consist of strips may be improved by the use of the Strip Squeeze algorithm.

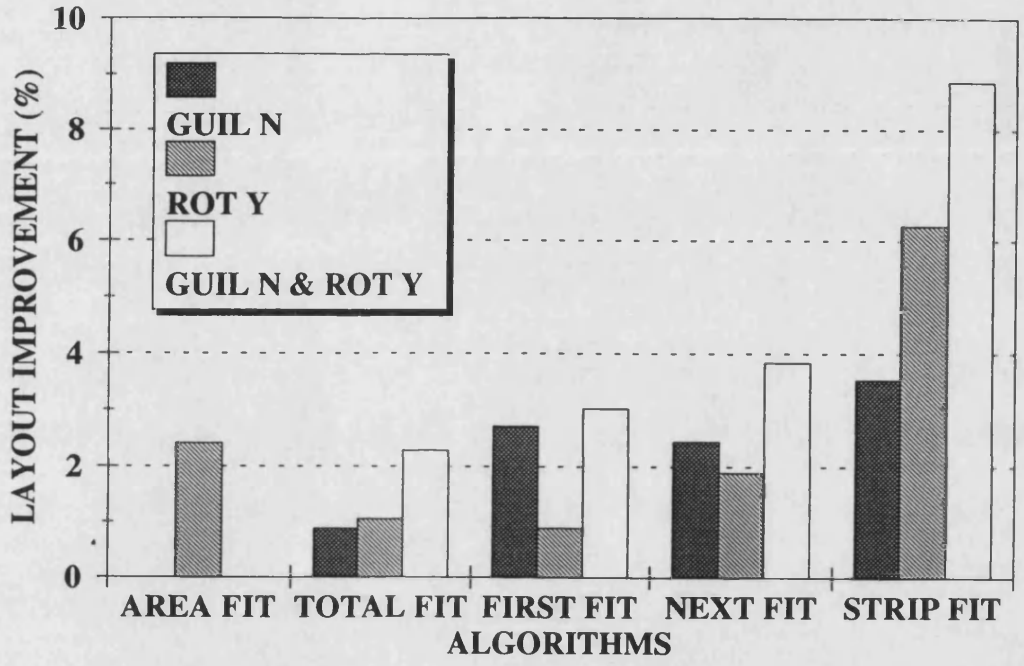


Figure 6.2.
The performance variation of the algorithms with each layout constraint.

The Strip Squeeze algorithm normally recovers a near identical sheet area when applied to the First Fit and Next Fit algorithms, as these algorithms are similar. The First Fit algorithm incurs less waste in its layout than the Next Fit algorithm. Thus the improvement in layout efficiency given by the Strip Squeeze algorithm is proportionally larger with the First Fit algorithm than with the Next Fit algorithm. The greatest layout improvement is achieved by the Strip Squeeze algorithm when it is used to improve the layout of the Strip Fit algorithm. This is because, after relaxation of the strip acceptance tolerances, the constituent parts of the strips produced will generally have a greater disparity in their widths and thus greater potential for improvement by 'squeezing'. The improvement achieved by the Strip Squeeze algorithm when applied to the Total Fit algorithm is considerably lower than when

applied to the other algorithms. This is due to the tight strip acceptance tolerances and also to some of the layout being generated by the Area Fit algorithm, which cannot be improved by 'squeezing'.

The performance of each algorithm is improved by allowing part rotation. The First Fit and Next Fit algorithms can only place the parts in a pre-determined orientation, so when the parts are allowed to rotate they are orientated with their largest dimension in the X axis. The resulting reduction in the average Y dimension of the parts list often allows longer strips to be formed and thus less waste incurred at the ends of the strips. As the First Fit algorithm places parts into these areas whenever possible a less significant improvement through part rotation is achieved. The Strip Fit, Area Fit and Total Fit algorithms can consider each part in both orientations which increases the range of part dimensions which can make good fits and thus the efficiency of the layout. The Strip Fit algorithm shows the greatest improvement of all the algorithms when the fixed part orientation constraint is removed. This is because the algorithm does not need to have the strip acceptance tolerances relaxed until much later in the nesting process, due to the greater choice of part dimensions, thus improving the overall layout efficiency.

Another perspective is that the more efficient a constrained layout is, the less potential there is for improving the layout by removing either of the constraints. This accounts for the large improvements achieved with the Strip Fit and Next Fit algorithms and the lesser improvements with the Total Fit algorithm. An exception to this is the Area Fit algorithm which shows an improvement of more than 2% when part rotation is permitted. This is exceptional in view of the efficiency of the constrained layout.

This also explains why the layout improvement achieved by relaxing both constraints together is less than the sum of each improvement obtained by relaxing each constraint individually. This is because once the layout has been improved by the relaxation of one constraint, there is less potential for improving the layout by relaxing the other constraint. The only exception to this is the Total Fit algorithm, where permitting part rotation allows a greater proportion of the layout to be formed by the Strip Fit element

of the algorithm before nesting is handed over to the Area Fit element. As the part of the layout created by the Area Fit element cannot be improved by the Strip Squeeze algorithm, permitting part rotation increases the proportion of the layout which can be improved and thus the improvement achieved.

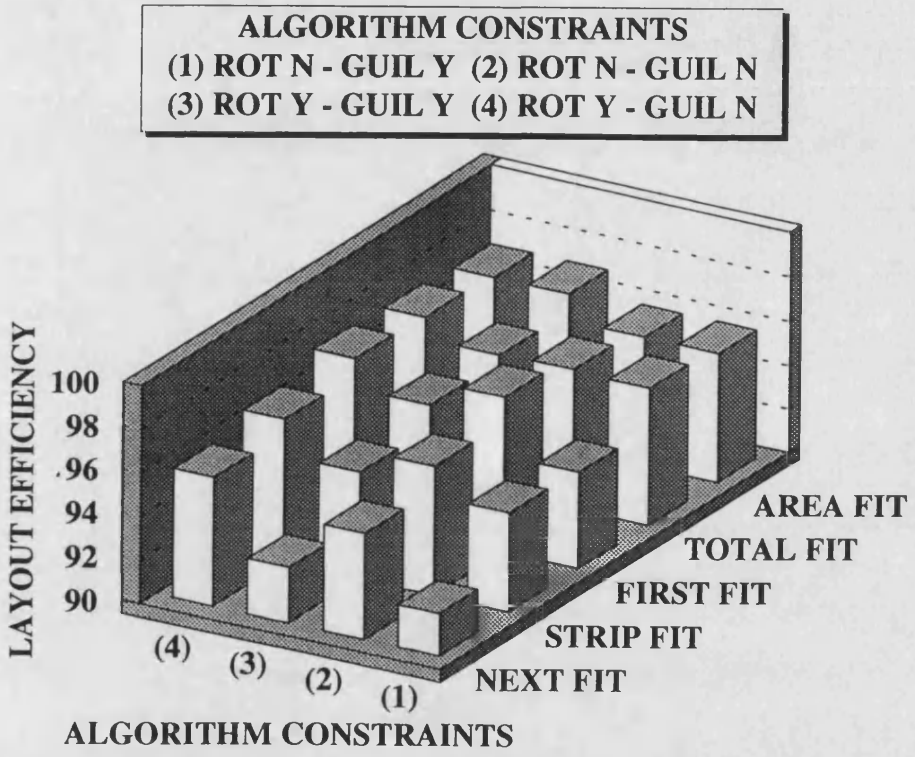


Figure 6.3.
The best layout efficiencies for all constraint variants of each algorithm.

The best layout efficiencies achieved by each algorithm for each constraint variant (across the range of parts lists and sheet sizes used) are shown in Figure 6.3 and are ordered in increasing efficiency from front to back. The algorithms at the front show more significant differences in 'best' layout efficiency due to constraint differences, which leads to the greater potential for improvement. When no constraints are applied to the layout (constraint 4 in the figure) there is little difference between the five algorithms. The application of the guillotine cut constraint, (1) and (3) in the figure, gives a considerable decrease in the best layout efficiency achieved by the Next Fit,

First Fit and Strip Fit algorithms. Thus the Strip Squeeze algorithm can be seen to improve the most efficient guillotine cut constrained layouts. In general, allowing part rotation does not significantly increase the efficiency of the best layouts produced.

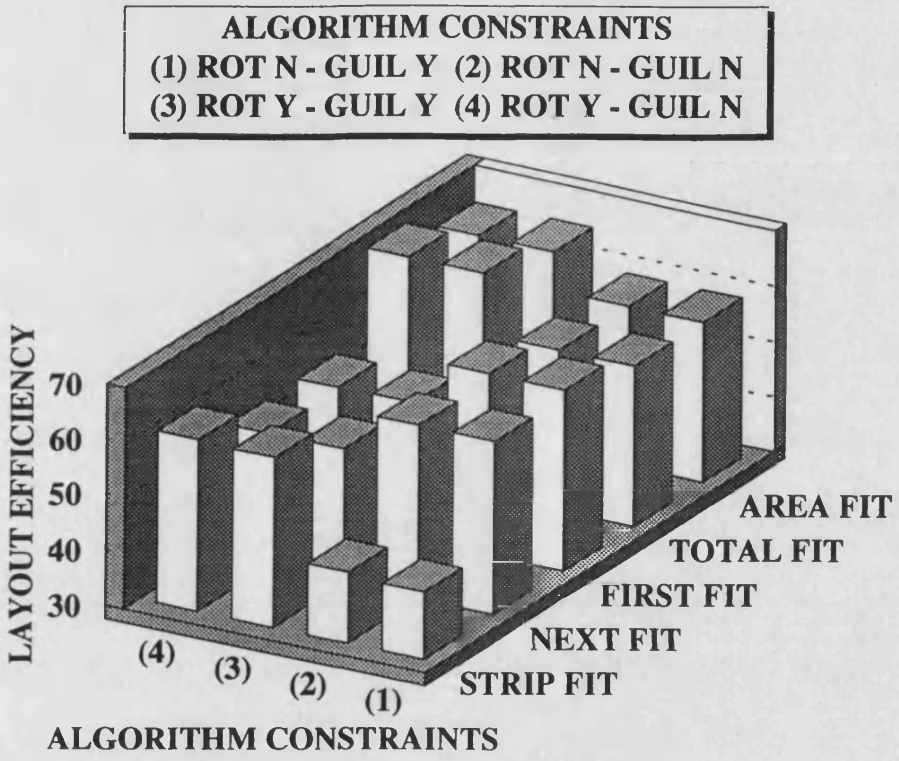


Figure 6.4.
The worst layout efficiencies for all constraint variants of each algorithm.

The worst layout efficiencies achieved by each algorithm for each constraint variant are shown in Figure 6.4. When the parts are not permitted to rotate the nesting efficiencies of worst layouts produced by the Strip Fit, Area Fit and Total Fit algorithms are considerably lower than when part rotation is permitted. This case is reversed for the Next Fit and First Fit algorithms. All these worst performances were achieved when placing a small list (10 parts) onto a relatively large sheet. The performances of the Next Fit and First Fit algorithms is worse with part rotation as one short wide strip is formed which increases the area at the end of the sheet which cannot be included in the 'Practical Efficiency'. However allowing part rotation is still

an advantage to the other three algorithms. It must be noted that all algorithms are designed to suit the most common operating circumstances and a considerable deterioration in performance is expected when an unusual nesting problem is tackled.

For the best layout efficiency values, only the Area Fit algorithm's performance was significantly affected by permitting parts to rotate. The relaxation of the guillotine cut constraint gave a significant difference in the performance of the other four algorithms. The relaxation of the guillotine cut constraint has little or no effect on worst layout efficiency performance of all the algorithms, however the worst case performances are affected by permitting parts to rotate. The worst case performances of the Strip Fit, Area Fit and Total Fit algorithms are improved by permitting part rotation. However this effect is reversed with the other two algorithms which show a less efficient worst case layout when part rotation is permitted.

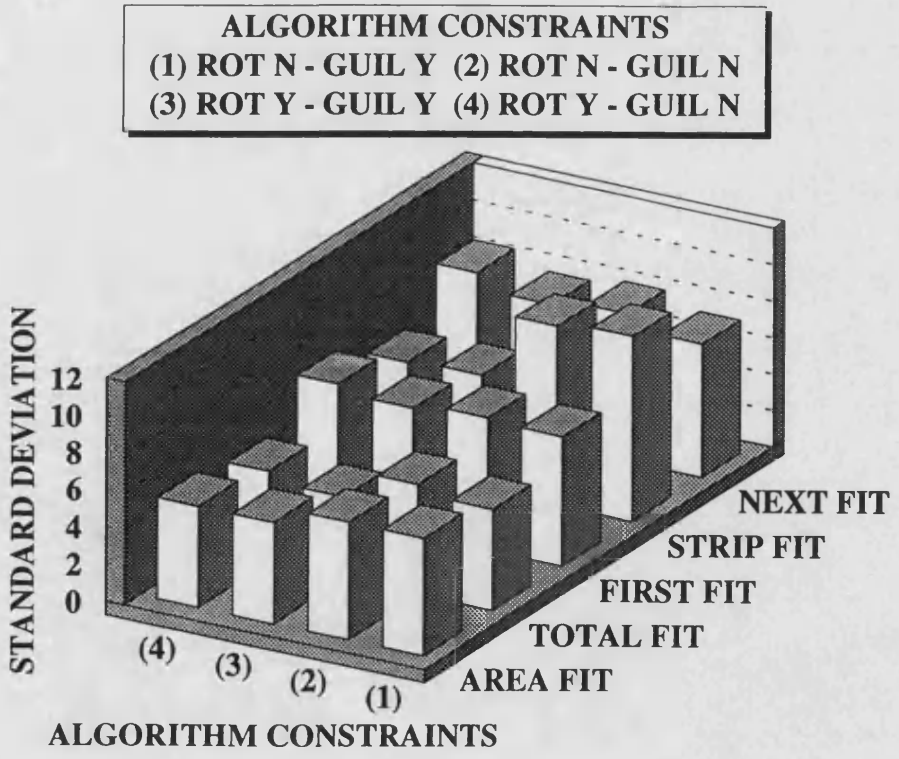


Figure 6.5.
The standard deviations for all constraint variants of each algorithm.

The standard deviations in the layout efficiency values for each algorithm with each constraint variant are shown in Figure 6.5. The Area Fit and the Total Fit have much lower standard deviations than the other three algorithms. This reflects their consistent performance and relatively good 'worst case' layout efficiency. The other three algorithms have much larger standard deviations which reflects their inability to create efficient layouts when presented with awkward parts lists and sheet sizes.

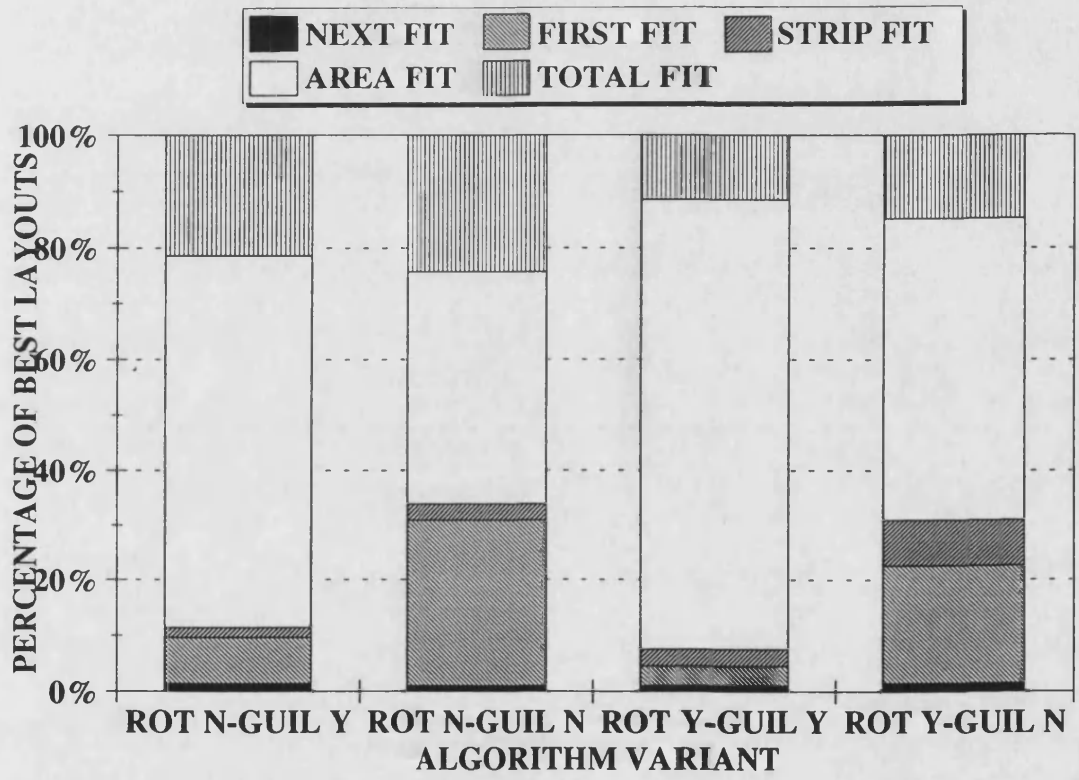


Figure 6.6.
The percentage of 'best layouts' created by each algorithm under each constraint variant.

So far the algorithms have been evaluated by considering the actual layout efficiencies achieved, rather than which algorithm is producing the most efficient layout for each combination of parts list and sheet size. This measure is far more important in deciding which algorithm is the best to use. The layouts which conform to each of the four constraint variants have been considered separately and the percentage of the

best layouts created by each algorithm is shown in Figure 6.6. If two or more algorithms produced the best layout efficiency they are each awarded a proportion of the best layout. This is why the Next Fit algorithm appears, even though the First Fit algorithm will always equal or better its layout. In all such cases the First Fit algorithm will have replicated the Next Fit algorithm's layout and will also have been awarded a proportion of the best layout.

The values given in Figure 6.6 support the earlier results. In all four constraint situations the Area Fit algorithm creates the greatest number of best layouts and the Next Fit algorithm creates the least number of best layouts. The Strip Fit algorithm creates the second lowest number of best layouts in all constraint situations. If the guillotine cut constraint is imposed (GUIL Y) the Total Fit algorithm creates the second highest number of best layouts and the First Fit algorithm creates the third highest number of best layouts. If the guillotine cut constraint is not imposed (GUIL N) the First Fit algorithm creates more best layouts than the Total Fit algorithm. This is because the Strip Squeeze algorithm, introduced in this work, can improve the First Fit algorithm's layout more than the Total Fit algorithms layout. The number of best layouts produced by the Strip Fit algorithm is increased considerably when all the layout constraints are removed (GUIL N - ROT Y). To allow a strictly accurate comparison to be made between the part placement algorithms developed in this research and the benchmark algorithms (First Fit and the Next Fit) the performance improvement achieved on these by the new Strip Squeeze algorithm should not be included. Thus only the guillotine cut constrained layouts (ROT N - GUIL Y and ROT Y - GUIL Y) are considered. In the former constraint situation, with part rotation not permitted, the benchmark algorithms only create 10% of the best layouts. With part rotation allowed for all algorithms this value drops to 5%. Thus the new placement algorithms (Strip Fit, Area Fit and Total Fit) are a considerable improvement on the benchmark placement algorithms and in most cases the new Strip Squeeze algorithm will improve any guillotine cut constrained strip layout by a significant amount.

Statistical Testing of the Results

Because the mean efficiency values achieved by many of the algorithms are similar, it needs to be established whether or not the new algorithms are significantly better than the benchmark algorithms. In every case the output of each algorithm has a histogram with a negatively skewed distribution (Table 6.1) and therefore the Mann-Whitney significance test was used, as it does not assume the data has a normal distribution. As the Next Fit algorithm can never give a better layout than the First Fit algorithm it is omitted from the Mann-Whitney test. The Strip Fit algorithm is intended to operate without tolerance relaxation as it is used in the Total Fit algorithm and thus the Strip Fit algorithm was not tested on its own. The Strip Squeeze algorithm gives considerable improvements in the layout efficiencies and would blur the comparison between them. Therefore only guillotine cut constrained layouts were considered when evaluating these three placement algorithms (Next Fit, Area Fit and Total Fit). The Strip Squeeze algorithm does not need to be statistically tested as it can only improve a layout (there are occasional exceptions to this with the Total Fit algorithm, where squeezing the strips results in an increased remaining sheet area which is subsequently less efficiently nested by the Area Fit algorithm).

To keep the test within reasonable bounds a random sample of 20 of the layout data sets was used from the main group of 189. The critical 'U' value to compare two sets of 20 values for a One-Tailed Mann-Whitney test at α=0.01 (99% confidence) is 114. The critical value at α=0.05 (95% confidence) is 138. The two sets of data are significantly different if the 'U' value produced is less than the critical value. The 'U' values obtained and the confidence level achieved for each test is shown in Table 6.3.

The Area Fit algorithm has been shown to be significantly better than the First Fit algorithm (greater than 99% confidence) and Total Fit algorithm (greater than 95% confidence) in both of the situations tested. This supports the conclusions drawn from the performance data examined earlier in this section. The Total Fit algorithm is only significantly better than the First Fit algorithm when the parts are held in a fixed

orientation. However this is because in this situation a greater proportion of the parts list will be placed by the Area Fit element of the system, which generally creates a more efficient layout.

PART ROT.	ALGORITHMS BEING COMPARED (FORMER BETTER THAN LATTER)	'U' VALUE	CONFIDENCE LEVEL
N	AREA FIT AND FIRST FIT	97.0	99%
	TOTAL FIT AND FIRST FIT	132.0	95%
	AREA FIT AND TOTAL FIT	136.5	95%
Y	AREA FIT AND FIRST FIT	58.0	99%
	TOTAL FIT AND FIRST FIT	169.5	BELOW 95%
	AREA FIT AND TOTAL FIT	71.5	99%

Table 6.3. Mann-Whitney test results for overall algorithm performance.

6.3 Effect of Part List Size

Purpose of the Test

If an algorithm is to place a list of parts, the number of parts within the list may affect the efficiency of the layout produced. The efficiency of the layout would be expected to generally increase with an increase in list size for three reasons. A greater number of parts should result in a greater choice of part to place in the early stages of nesting. A greater number of parts will result in more parts with a similar dimension to form efficient strips. Finally, when calculating the 'Practical Efficiency', waste is incurred at the end of the layout by discarding any small remaining areas. In a large layout this will be a smaller proportion of the total nested area and therefore has less effect.

Parts Lists and Sheet Sizes used in this Test

A number of parts lists were generated each of which contained a different number of parts. Other parameters of the parts lists, such as the average part area and average part aspect ratio, were made reasonably common to all the lists which were compared. Three sets of five parts lists were generated. For all sets the constituent lists contain 10, 20, 40, 80 and 160 parts. The average area and the average dimensions in all the lists of each set of parts were exactly the same. The maximum and minimum dimensions varied only slightly and the average aspect ratio of the parts was held to a tight tolerance for all the parts lists in each set.

The actual parameter values for each parts list are shown in Table 6.4. Each of these parts lists is to be nested on three sheet sizes, 2000mm by 1000mm, 2500mm by 1500mm and 3000mm by 2000mm. This resulted in 45 individual test runs. The parts of the first list in set 1 (SIZ11) were generated by the Random Method described in Section 4.5. The other four parts lists within this set (SIZ12-SIZ15) were created by duplicating the parts from the first list the required number of times ie. the fourth list (SIZ14) contains 80 parts and therefore 8 off of each part in the first list. Using so

many identical parts does not set as challenging a nesting problem as a completely random set of parts. Thus the other two sets of parts lists (SIZ21-SIZ35) were generated to contain lists with common characteristics, but not multiples of identical parts. To generate these 10 lists the 'Fixed Area and Aspect Ratio Method' (Section 4.5) was used. This method operates by repeatedly splitting a given rectangular area into a groups of 10 parts to create the list size required (in multiples of 10). This process was repeated until the required average aspect ratio was achieved.

SET	FILENAME OF PARTS LIST	NUMBER OF PARTS	AVERAGE DIMENSION	MAXIMUM DIMENSION	AVERAGE ASPECT RATIO
1	SIZ11	10	502mm	868mm	0.402
	SIZ12	20	502mm	868mm	0.402
	SIZ13	40	502mm	868mm	0.402
	SIZ14	80	502mm	868mm	0.402
	SIZ15	160	502mm	868mm	0.402
2	SIZ21	10	300mm	476mm	0.585
	SIZ22	20	300mm	488mm	0.585
	SIZ23	40	300mm	499mm	0.581
	SIZ24	80	300mm	499mm	0.581
	SIZ25	160	300mm	500mm	0.585
3	SIZ31	10	215mm	397mm	0.444
	SIZ32	20	215mm	387mm	0.444
	SIZ33	40	215mm	391mm	0.444
	SIZ34	80	215mm	398mm	0.444
	SIZ34	160	215mm	397mm	0.444

Table 6.4. Details of parts lists used to evaluate the effect of parts list size on the efficiency of the layout produced.

Results of the Test

The percentage efficiency achieved by each algorithm in relation to the size of the parts list is given in Figures 6.7 to 6.11. All five graphs have been plotted in the same range to allow easy comparison. Each percentage efficiency value given (Table 6.5) is the average of nine individual tests, due to three parts lists of each size each being placed on three different sheet sizes. The effect of parts list size is isolated by the other parameters in all five sets of nine tests being identical.

ALGO.	ROT	GUIL	PARTS LIST SIZE				
			10	20	40	80	160
NEXT FIT	N	Y	70.5%	74.0%	76.9%	77.2%	77.6%
NEXT FIT	N	N	77.0%	79.4%	78.9%	78.6%	77.8%
NEXT FIT	Y	Y	74.0%	77.6%	76.9%	81.2%	79.8%
NEXT FIT	Y	N	78.1%	80.6%	78.6%	81.9%	80.2%
FIRST FIT	N	Y	71.7%	77.5%	82.9%	85.2%	86.3%
FIRST FIT	N	N	78.7%	83.3%	85.8%	86.0%	86.8%
FIRST FIT	Y	Y	77.5%	83.9%	83.5%	87.3%	87.5%
FIRST FIT	Y	N	81.8%	87.1%	85.3%	87.8%	87.8%
STRIP FIT	N	Y	67.3%	73.8%	80.1%	83.2%	85.7%
STRIP FIT	N	N	73.5%	77.6%	84.3%	84.6%	87.7%
STRIP FIT	Y	Y	78.5%	83.1%	85.4%	88.7%	90.8%
STRIP FIT	Y	N	82.6%	85.8%	88.9%	90.6%	92.5%
AREA FIT	N	Y&N	77.5%	82.6%	86.6%	89.1%	91.9%
AREA FIT	Y	Y&N	78.0%	86.0%	90.1%	92.9%	94.3%
TOTAL FIT	N	Y	77.5%	82.7%	87.4%	88.9%	90.7%
TOTAL FIT	N	N	77.5%	82.7%	87.4%	89.8%	91.4%
TOTAL FIT	Y	Y	79.8%	85.4%	89.3%	90.6%	92.1%
TOTAL FIT	Y	N	79.8%	85.5%	90.5%	92.0%	93.2%

Table 6.5. Algorithm performance values with different sized parts lists.

The curves of the constraint variants of the Next Fit algorithm (Figure 6.7) are erratic, but generally show an improvement in nesting efficiency as the size of the parts list increases. The performances of the two variants which are constrained by the guillotine cut constraint (GUIL Y) are particularly poor when nesting only 10 parts. Such a small parts list will form strips which taper considerably, and thus can be markedly improved by the removal of the guillotine cut constraint ie. the application of the Strip Squeeze algorithm. A large parts list forms strips which taper less, reducing the effect of this constraint. The re-orientation of the parts to have their larger dimension in the X axis (ROT Y) only becomes a clear benefit with the larger parts list sizes of 80 and 160 parts. Thus with a small parts list relaxing the guillotine cut constraint is of greatest benefit and with a large parts list allowing part rotation is of greatest benefit.

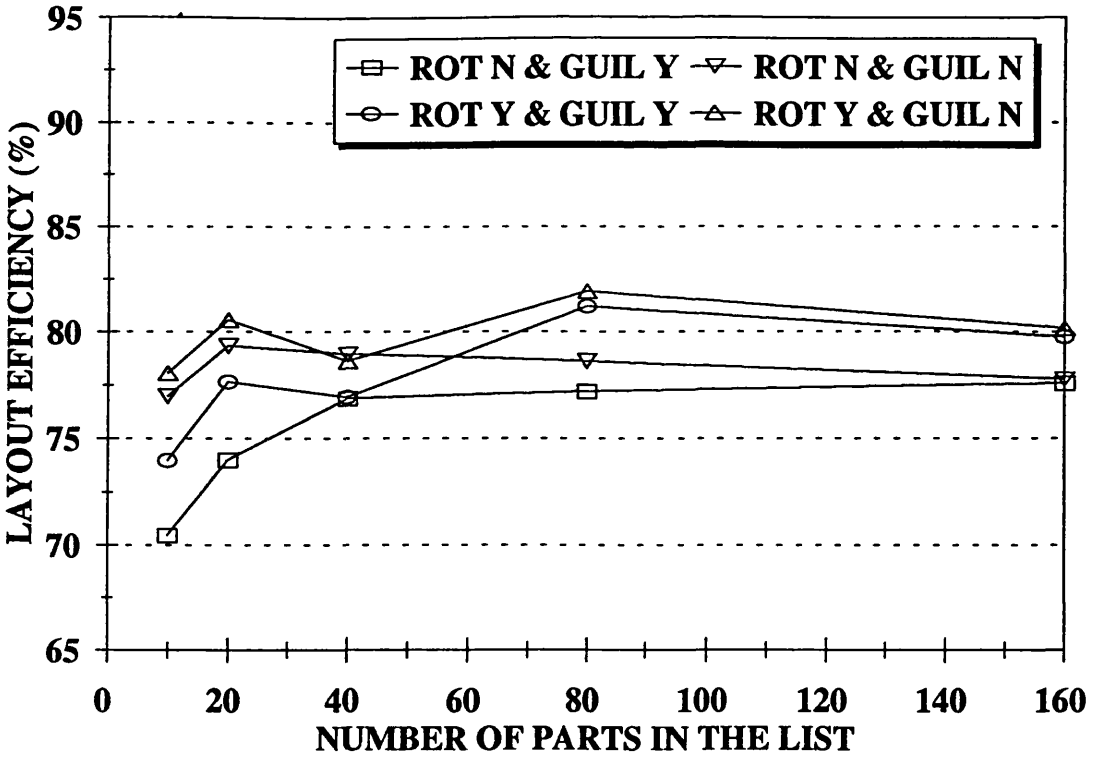


Figure 6.7.
The performance variation of the Next Fit algorithm with parts list size.

With a small parts list the First Fit algorithm (Figure 6.8) generates layouts with a slightly better efficiency than those produced by the Next Fit algorithm. However, the performance of this algorithm increases more clearly with the increase in parts list size. Thus the layout efficiencies achieved by the First Fit algorithm with a large parts list are much better than those of the Next Fit algorithm. As with the Next Fit algorithm, the variants which conform the guillotine cut constraint perform badly with a small list size. Fixing each part's larger dimension on the X axis (ie. part rotation) seems to be generally of benefit in this algorithm, however this benefit is not as great as that seen with the Next Fit algorithm when nesting large parts lists.

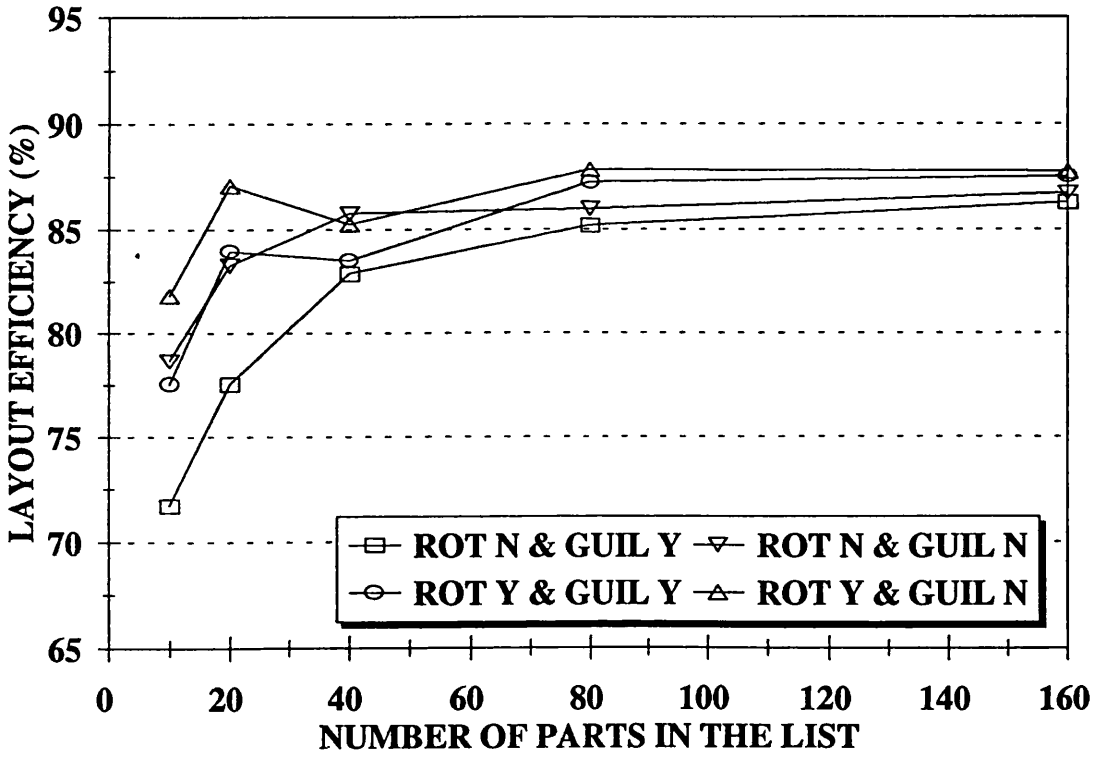


Figure 6.8.
The performance variation of the First Fit algorithm with parts list size.

The performance of the Strip Fit algorithm in relation to parts list size (Figure 6.9) shows the greatest differences of all of the algorithms tested. The worst average layout efficiency of 67.3% is seen with the smallest list of 10 parts, which occurred with the most constrained variant (ROT N & GUIL Y). The performance of this

variant is radically improved in average layout efficiency to 85.7% with the largest list size of 160 parts. The four constraint variants can clearly be ranked in performance order and the improvement seen with a large parts list is again considerable.

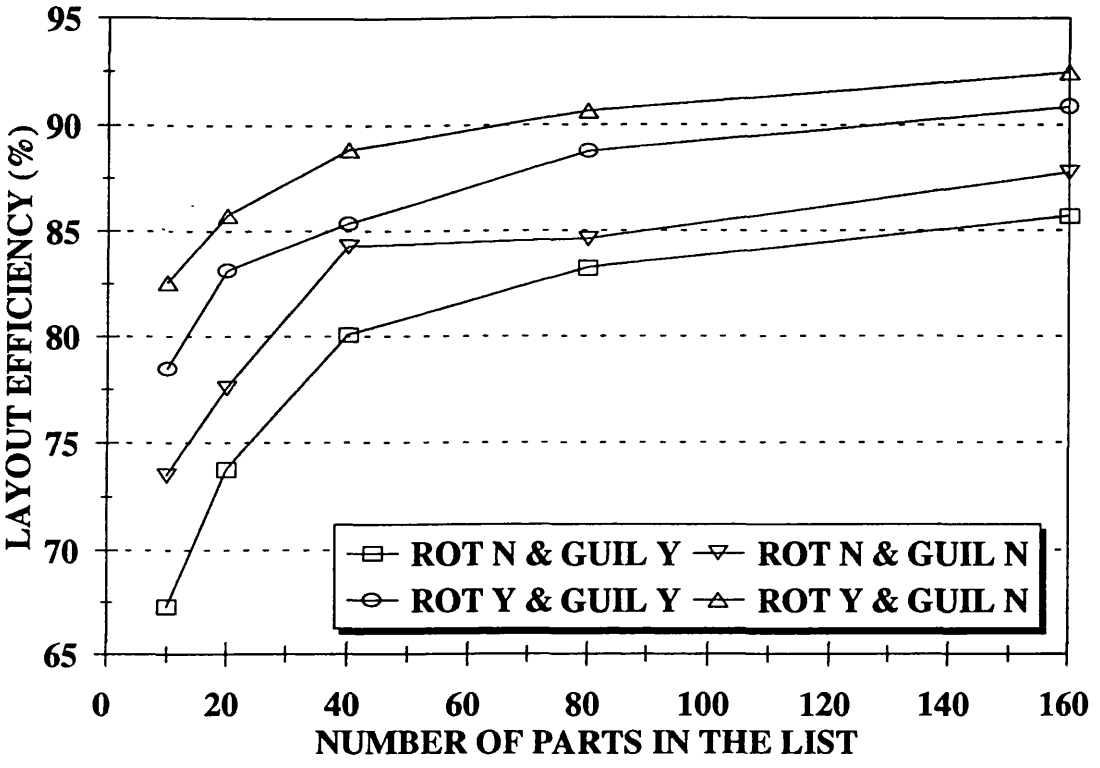


Figure 6.9.
The performance variation of the Strip Fit algorithm with parts list size.

Only two variant curves are plotted on the Area Fit graph (Figure 6.10) as this algorithm always produces layouts which conform to the guillotine cut constraint. There is a clear improvement in layout efficiency with the increase in parts list size due to the greater choice of parts to place. The freedom to rotate parts into a chosen orientation gives a good improvement in layout efficiency (2% to 4%) with all but the smallest parts list.

Finally the Total Fit algorithm (Figure 6.11) also shows a clear improvement in layout performance with an increase in parts list size, again due to the greater choice of parts to place. Removal of the guillotine cut constraint gives no improvement in layout

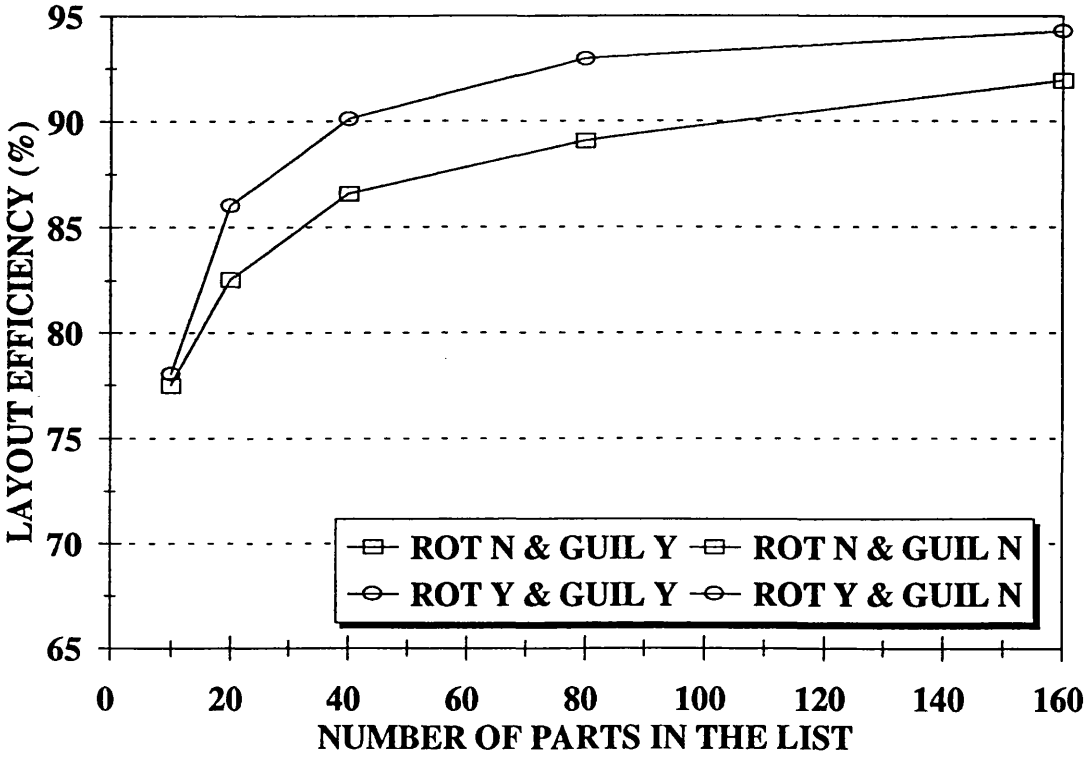


Figure 6.10.
The performance variation of the Area Fit algorithm with parts list size.

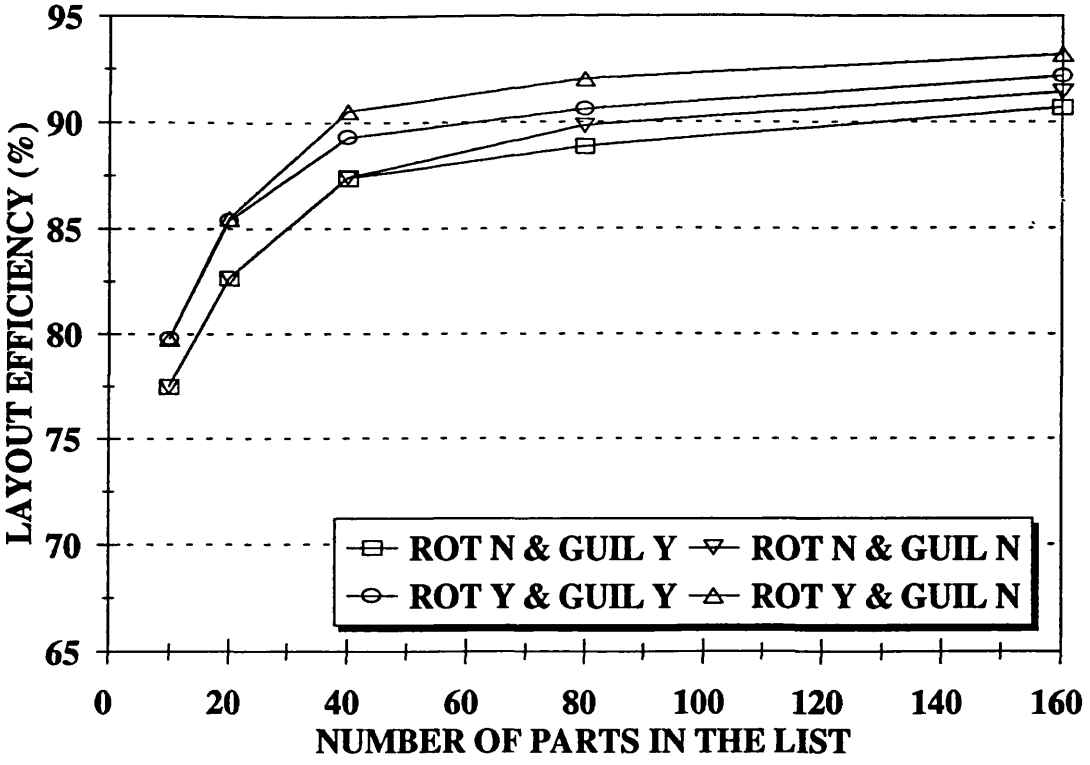


Figure 6.11.
The performance variation of the Total Fit algorithm with parts list size.

efficiency with a small parts list. This is because the Strip Fit element of the algorithm is not able to produce two strips of parts before handing over nesting to the Area Fit algorithm. Thus the layout does not contain two strips to 'squeeze' together preventing any layout improvement being made from the relaxation of the guillotine cut constraint. The removal of this constraint allows a small, but significant improvement in layout efficiency with a larger parts list. When part rotation is permitted, improvement is seen with smaller parts lists. Allowing part rotation gives the Strip Fit algorithm a greater choice of dimension combinations, which in turn leads to more strips being formed from each list of parts before the Area Fit element takes over nesting.

Statistical Testing of the Results

Because the layout efficiencies achieved by the algorithms do not change vastly with changes in the parts list size, it needed to be established whether or not these differences in layout efficiency were significant. The results do not seem to conform to a normal distribution and therefore the Mann-Whitney test has been used to establish significance levels. To reduce the testing, only the variant with least difference in performance between the largest and smallest parts list was tested. If a significant difference was found in the performance of this algorithm with a list of 10 parts and a list of 160 parts, this should be true for the other variants. If this did not display a significant difference at the 95% confidence level then the other constraint variants of the algorithm were also tested.

There are nine performance values for each parts list size. The critical 'U' value to compare two sets of nine values for a One-Tailed Mann-Whitney test at α=0.01 (99% significance) is 14. The critical value at α=0.05 (95% significance) is 21. The two sets of data are significantly different if the 'U' value produced is less than the critical value. The values obtained are shown in Table 6.6.

ALGORITHM	PART ROT	GUIL CUT	'U' VALUE	CONFIDENCE LEVEL
NEXT FIT	N	N	38.0	BELOW 95%
FIRST FIT	N	N	19.0	95%
STRIP FIT	Y	N	6.0	99%
AREA FIT	N	Y&N	2.0	99%
TOTAL FIT	Y	Y	3.0	99%
NEXT FIT	N	Y	17.0	95%
NEXT FIT	Y	Y	22.5	BELOW 95%
NEXT FIT	Y	N	34.0	BELOW 95%

Table 6.6. Mann-Whitney test results for effect of parts list size.

The first five rows in Table 6.6 show the 'U' values for the constraint variant of each algorithm (Figures 6.7 to 6.11) which has the least difference in performance between the largest and smallest list. From these we can be 99% confident that all the new algorithms (Strip Fit, Area Fit and Total Fit) will produce more efficient layouts when nesting lists of 160 parts rather than 10 parts. The First Fit algorithm 'U' value is within the threshold critical value for 95% confidence. The Next Fit algorithm's 'U' value is outside the 95% confidence level, requiring the other three variants to be tested. These values are given in the bottom three rows of Table 6.6 and only one variant's 'U' value falls within the 95% confidence level bounds. Thus the new algorithms benefit to a far greater extent from a large parts list than the benchmark algorithms. It can be assumed, in general terms, that nesting efficiency improves with part list size for all but the Next Fit algorithm.

6.4 Effect of Average Part Aspect Ratio

Purpose of the Test

If an algorithm is to place a list of parts, the average aspect ratio of parts within the list may affect the efficiency of the layout produced. The efficiency of the layout would be expected to generally decrease with an increase in average part aspect ratio for two reasons. A higher average part aspect ratio represents a greater range of part dimensions. This will make the creation of strips of parts with a similar X dimension and the matching of parts with spaces on the sheet more difficult. Secondly, the larger aspect ratio will result in some large part dimensions which, if they approach the size of one of the sheet dimensions, will be particularly unwieldy and difficult to place efficiently.

Parts Lists and Sheet Sizes used in this Test

A number of lists were generated each of which contained parts with different average aspect ratios. Also, the effect of part aspect ratio is clearer if the average part size (area) for every list is common. The parts lists were generated using the 'Fixed Aspect Ratio Method', described in Section 4.5. This method simultaneously creates a set of four parts lists with the same average part area. The average part aspect ratio in each list was held to the exact values of 1:1, 1:2.25, 1:4 and 1:9 by giving every part in each list the required average aspect ratio. Four sets of four parts lists with these average aspect ratios were generated for this test series. The maximum and minimum dimensions within each list varies in accordance with these aspect ratios. The actual values for each parts list are shown in Table 6.7. Each of the four parts lists (in each of the four sets) was nested by all the algorithms onto three sheet sizes; 3000mm by 2000mm, 4000mm by 2500mm and 5000mm by 3000mm, giving 48 individual test runs.

SET	FILENAME OF PARTS LIST	NUMBER OF PARTS IN LIST	AVERAGE DIMENSION	MAXIMUM DIMENSION	AVERAGE ASPECT RATIO
1	ASP11	150	385mm	660mm	1:1.00
	ASP12	150	417mm	990mm	1:2.25
	ASP13	150	481mm	1320mm	1:4.00
	ASP14	150	642mm	1980mm	1:9.00
2	ASP21	150	468mm	660mm	1:1.00
	ASP22	150	507mm	990mm	1:2.25
	ASP23	150	586mm	1320mm	1:4.00
	ASP24	150	781mm	1920mm	1:9.00
3	ASP31	100	202mm	324mm	1:1.00
	ASP32	100	219mm	486mm	1:2.25
	ASP33	100	253mm	648mm	1:4.00
	ASP34	100	337mm	972mm	1:9.00
4	ASP41	30	575mm	660mm	1:1.00
	ASP42	30	623mm	990mm	1:2.25
	ASP43	30	719mm	1320mm	1:4.00
	ASP44	30	958mm	1980mm	1:9.00

Table 6.7. Details of parts lists used to evaluate the effect of part aspect ratio on the efficiency of the layout produced.

Results of the Test

The performance of each algorithm in relation to the average aspect ratio of the parts nested is given in Figures 6.12 to 6.16. All five graphs have been plotted in the same range to allow easy comparison. Each point plotted is the average of twelve individual tests, due to four parts lists of each part aspect ratio (one from each set, eg. ASP11, ASP21, ASP31, ASP41) being placed on three different sheet sizes. The average performance values for each algorithm are given in Table 6.8. It must be noted that

parts with an aspect ratio of 1:1 are squares and therefore there will be no increase in layout efficiency when part rotation is permitted.

ALGORITHM	ROT	GUIL	PART ASPECT RATIO			
			1.00	2.25	4.00	9.00
NEXT FIT	N	Y	84.6%	82.2%	76.9%	71.7%
NEXT FIT	N	N	85.5%	84.3%	80.1%	76.0%
NEXT FIT	Y	Y	84.6%	84.9%	83.4%	73.2%
NEXT FIT	Y	N	85.5%	86.4%	85.1%	78.1%
FIRST FIT	N	Y	88.6%	86.7%	81.9%	75.5%
FIRST FIT	N	N	90.0%	88.8%	85.1%	80.3%
FIRST FIT	Y	Y	88.6%	87.8%	85.1%	74.9%
FIRST FIT	Y	N	90.0%	89.1%	86.9%	79.2%
STRIP FIT	N	Y	87.5%	78.8%	74.3%	56.0%
STRIP FIT	N	N	89.4%	82.7%	77.6%	59.3%
STRIP FIT	Y	Y	87.5%	88.3%	83.4%	72.9%
STRIP FIT	Y	N	89.4%	89.9%	86.1%	75.4%
AREA FIT	N	Y&N	91.6%	90.3%	89.3%	81.2%
AREA FIT	Y	Y&N	91.6%	91.6%	93.5%	87.8%
TOTAL FIT	N	Y	91.5%	88.7%	86.5%	82.4%
TOTAL FIT	N	N	92.0%	90.0%	87.4%	82.0%
TOTAL FIT	Y	Y	91.5%	91.5%	87.7%	86.2%
TOTAL FIT	Y	N	91.9%	92.4%	89.0%	85.2%

Table 6.8. Algorithm performance values with different average part aspect ratios.

All variants of the Next Fit algorithm (Figure 6.12) and First Fit algorithm (Figure 6.15) show an overall decrease in layout efficiency as the aspect ratio of the parts increases. For both algorithms the improvement in layout efficiency achieved by the Strip Squeeze algorithm in general increases with the increasing aspect ratio. This is

expected as the strips formed when nesting a list with a high average part aspect ratio will contain large disparities in the X dimensions of the member parts. This can be seen by comparing the strips in Figure 6.13 to those in Figure 6.14.

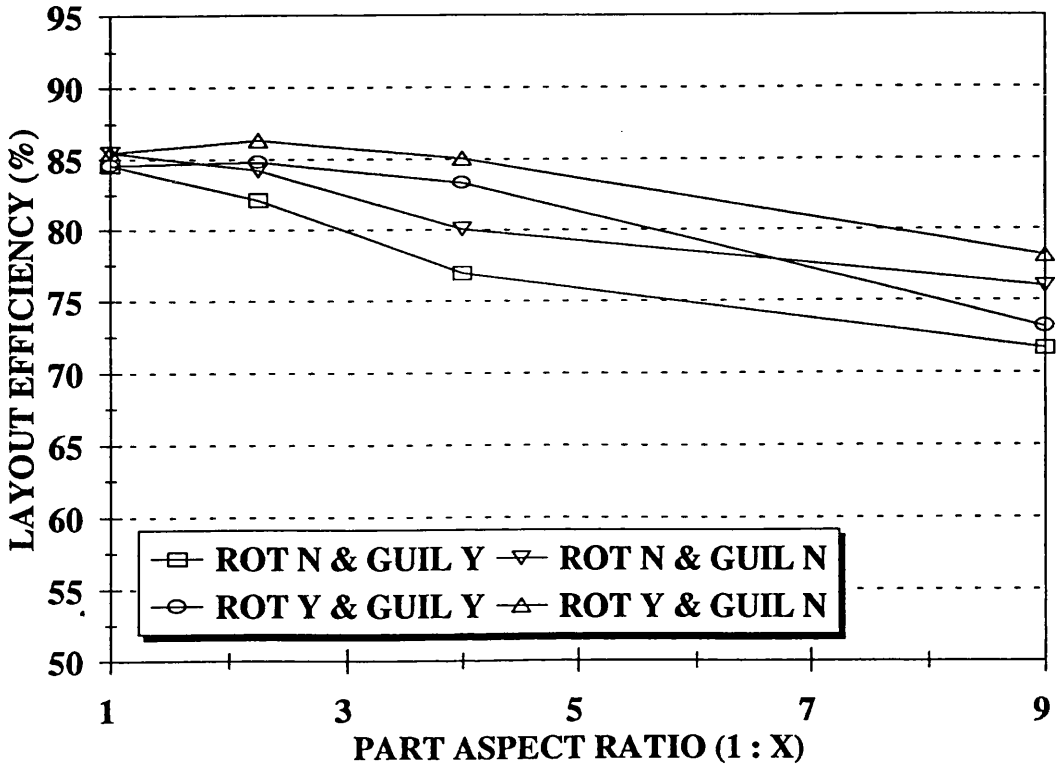


Figure 6.12.
The performance variation of the Next Fit algorithm with average part aspect ratio.

The layout efficiency achieved by the First Fit algorithm (Figure 6.15) shows a steady decrease in layout efficiency as the average part aspect ratio increases. This is due to the average width of the strips increasing, which incurs additional waste in two ways. Firstly, this will cause the 'end of sheet' waste in a multiple sheet layout to be, on average, larger. This waste is due to the next part to place not fitting the X dimension of the remaining sheet area and a new sheet being started (see Section 5.2). This can be seen clearly at the end of the first sheet in Figure 6.14 (this layout has been created by the Next Fit algorithm, but the feature is common to both algorithms). Secondly, the area at the end of the last strip will be wider, splitting the remaining area on the last sheet into two more evenly sized sections. This division reduces the 'Practical Efficiency' value which only considers the larger of these areas.

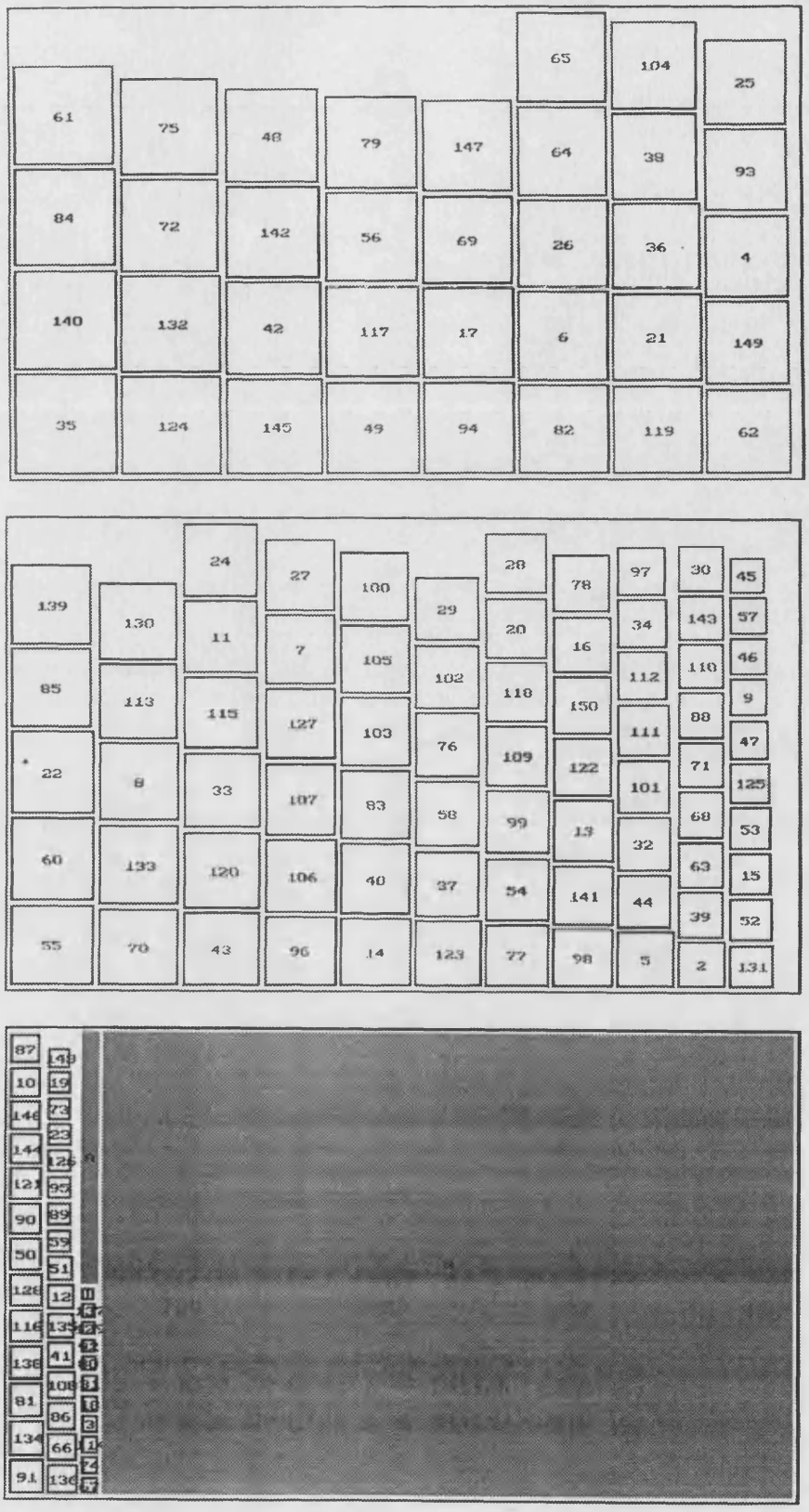


Figure 6.13.
A three sheet Next Fit algorithm layout, no Strip Squeeze, part rotation permitted (but redundant), aspect ratio of all parts 1:1.00.

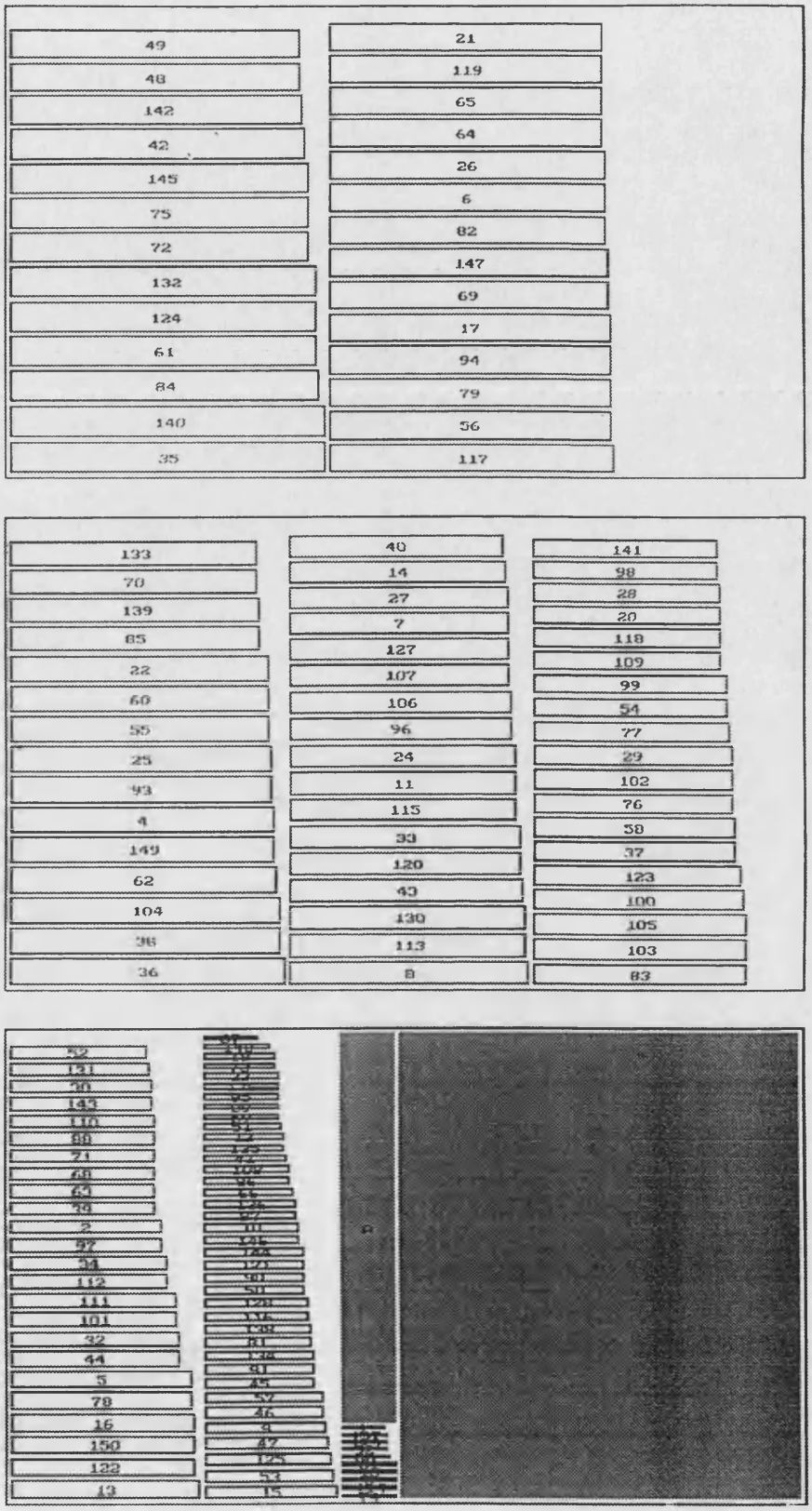


Figure 6.14.
A three sheet Next Fit algorithm layout, no Strip Squeeze, part rotation permitted, aspect ratio of all parts 1:9.00.

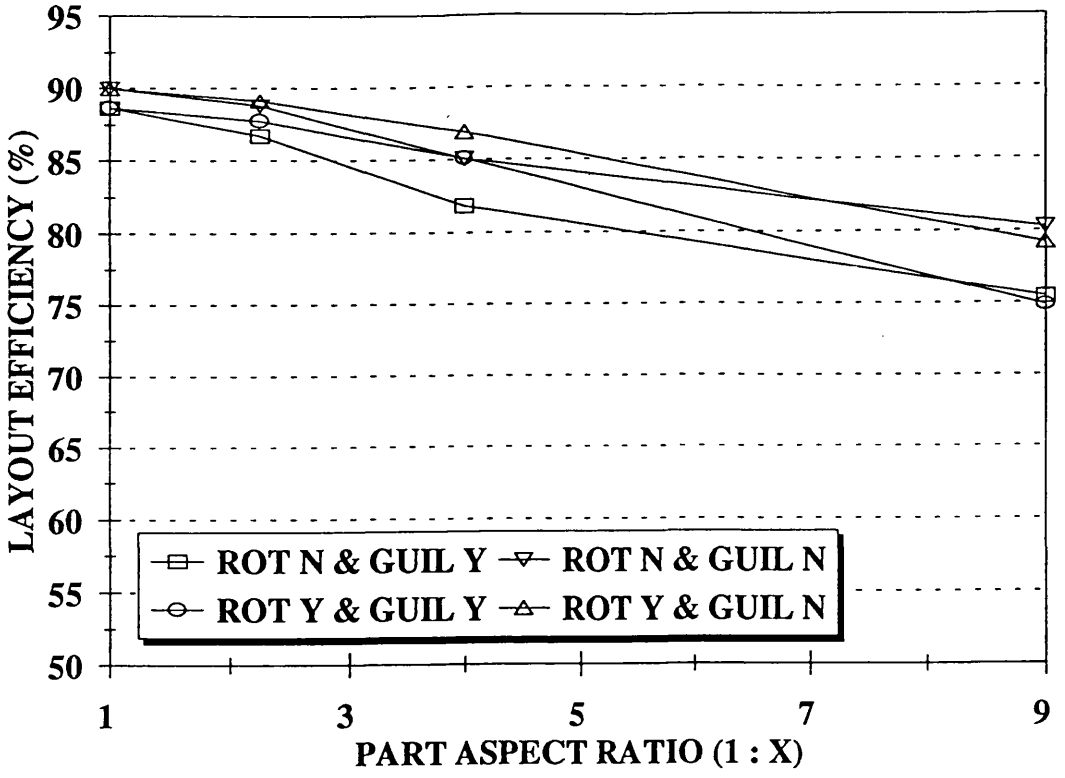


Figure 6.15.
The performance variation of the First Fit algorithm with average part aspect ratio.

When part rotation is permitted the Next Fit algorithm (Figure 6.12) shows a slight increase in layout efficiency as the aspect ratio of the parts increases from 1:1.00. This is due to the reduction in the Y dimensions of the parts allowing the creation of longer strips, which can be seen by comparing Figures 6.13 and 6.14. However as the aspect ratio increases further this saving is overtaken by the increasing 'end of sheet' and 'Practical Efficiency' losses described in the preceding paragraph.

The performance of the Strip Fit algorithm, shown in Figure 6.16, is reduced by a greater extent than any of the other algorithms by the increase in part aspect ratio. This results in it having the worst performance of all the algorithms when nesting a list of parts with high aspect ratios. As before, the guillotine cut constraint does not reduce the layout efficiency achieved to the same extent as when holding the parts in a fixed orientation. The performance of this algorithm with part rotation permitted initially increases as the aspect ratio increases from 1:1.00 ie. a list of square parts.

The rotation of square parts does not increase the possible strip permutations, however the ability to rotate non-square parts does. This is beneficial until the parts become of such an extreme aspect ratio that their unwieldy dimensions require considerable relaxation of the strip acceptance tolerances and thus reduce the efficiency of the layouts generated.

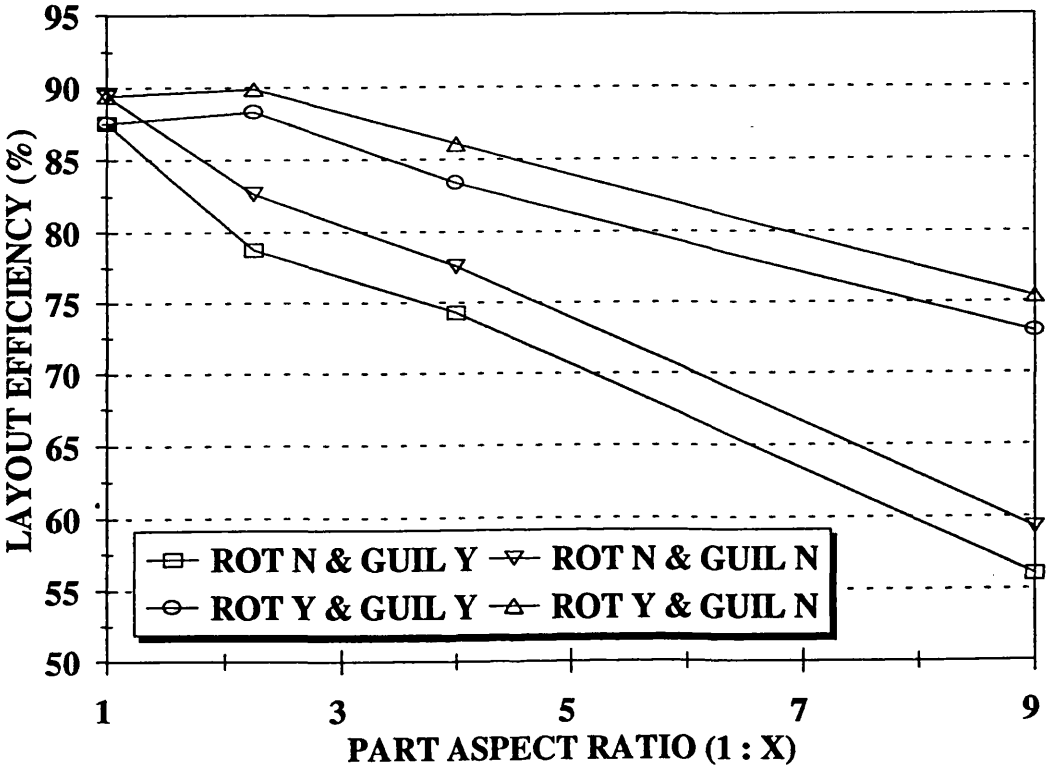


Figure 6.16.
The performance variation of the Strip Fit algorithm with average part aspect ratio.

When part rotation is not permitted the Area Fit algorithm's performance progressively drops with an increase in part aspect ratio, as shown in Figure 6.17. However, when part rotation is permitted the layout efficiency initially increases, which is expected as being able to rotate non-square parts increases the range of part dimensions which can be placed in either axis. However, as the part aspect ratio becomes more extreme this benefit is counteracted by the parts becoming unwieldy. The two lines diverge as the part aspect ratio increases, which indicates that the benefit of permitting part rotation increases with the increase in part aspect ratio. This algorithm performs considerably better than all the others when nesting parts with a high aspect ratio.

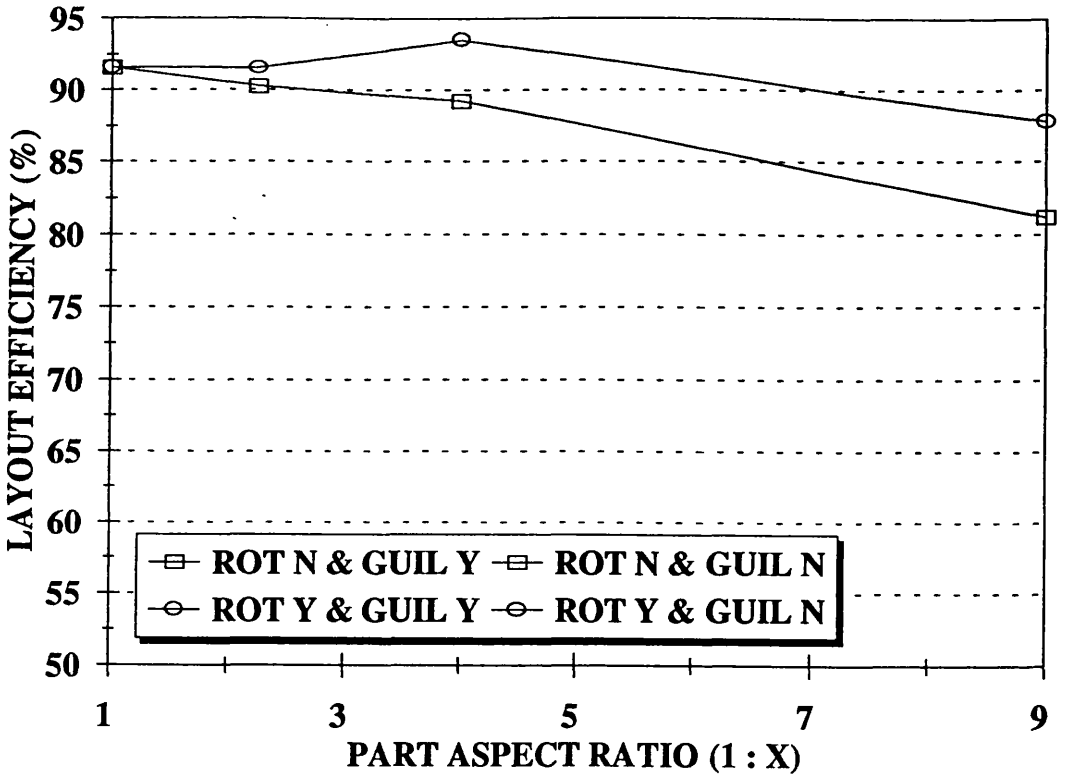


Figure 6.17.
The performance variation of the Area Fit algorithm with average part aspect ratio.

An example Area Fit layout of high aspect ratio (1:9.00) parts is shown in Figure 6.18. This is the same part list and sheet size used in the Next Fit layout shown in Figure 6.14. This layout is a good example of the robustness of the Area Fit algorithm to awkward nesting situations. It also demonstrates the element of chance in producing a layout with a good 'Practical Efficiency'. If the sheet had been fractionally larger or the list contained one part less, the third sheet would not have been started. Any algorithm's performance can suffer in this manner.

The Total Fit algorithm (Figure 6.19) creates layouts which are slightly less efficient than those of the Area Fit algorithm when placing parts with a high aspect ratio. The performance of this algorithm is also reduced by holding the parts with a high aspect ratio in a fixed orientation. With high aspect ratio parts the Strip Squeeze algorithm gives little improvement in the efficiency of layouts created by the Total Fit algorithm. This suggests that the Strip Fit element of the system places few strips before the remaining parts list is handed to the Area Fit algorithm for nesting.

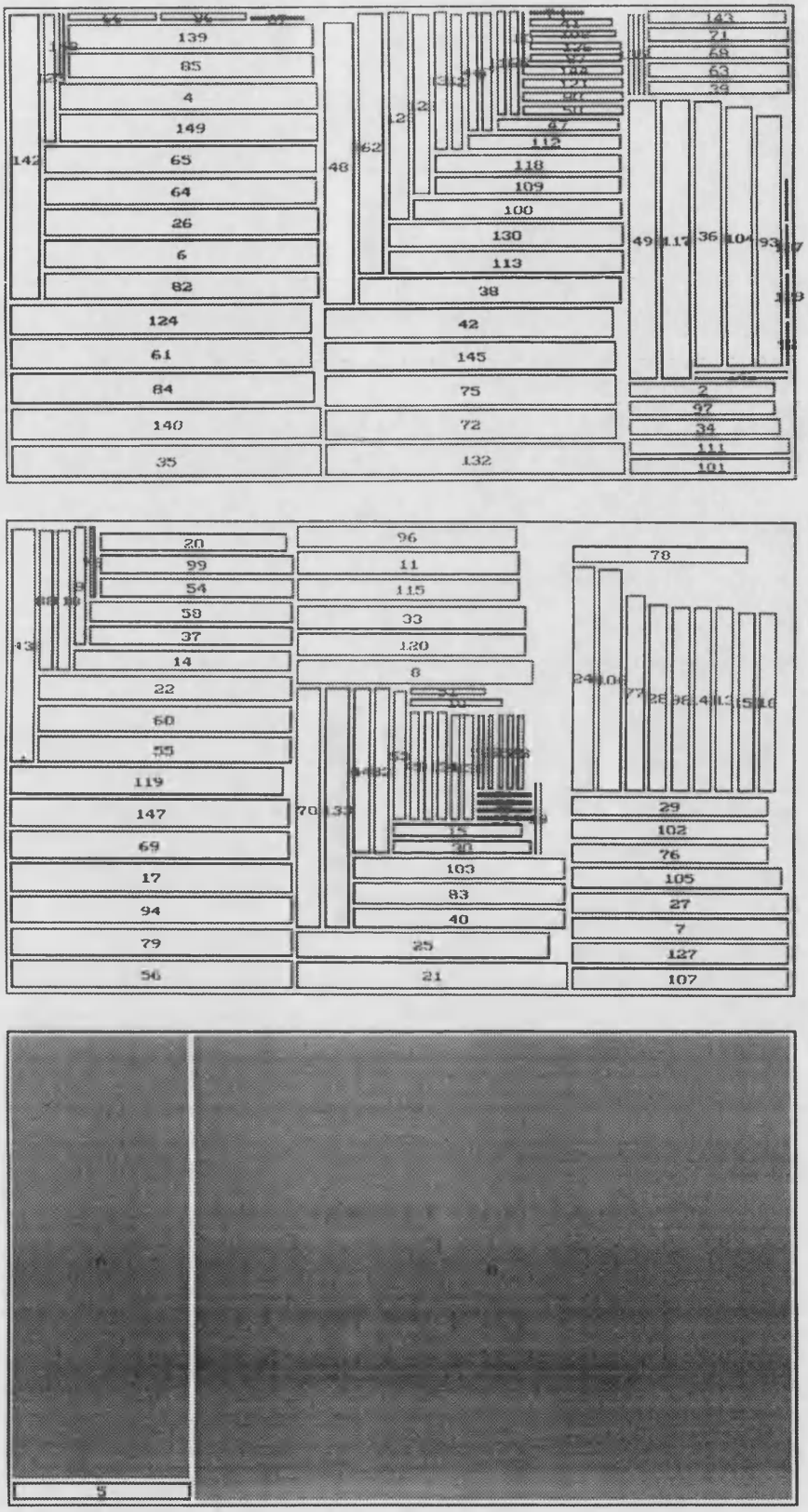


Figure 6.18.
A three sheet layout by the Area Fit algorithm (part rotation permitted) with an average part aspect ratio of 1:9.00.

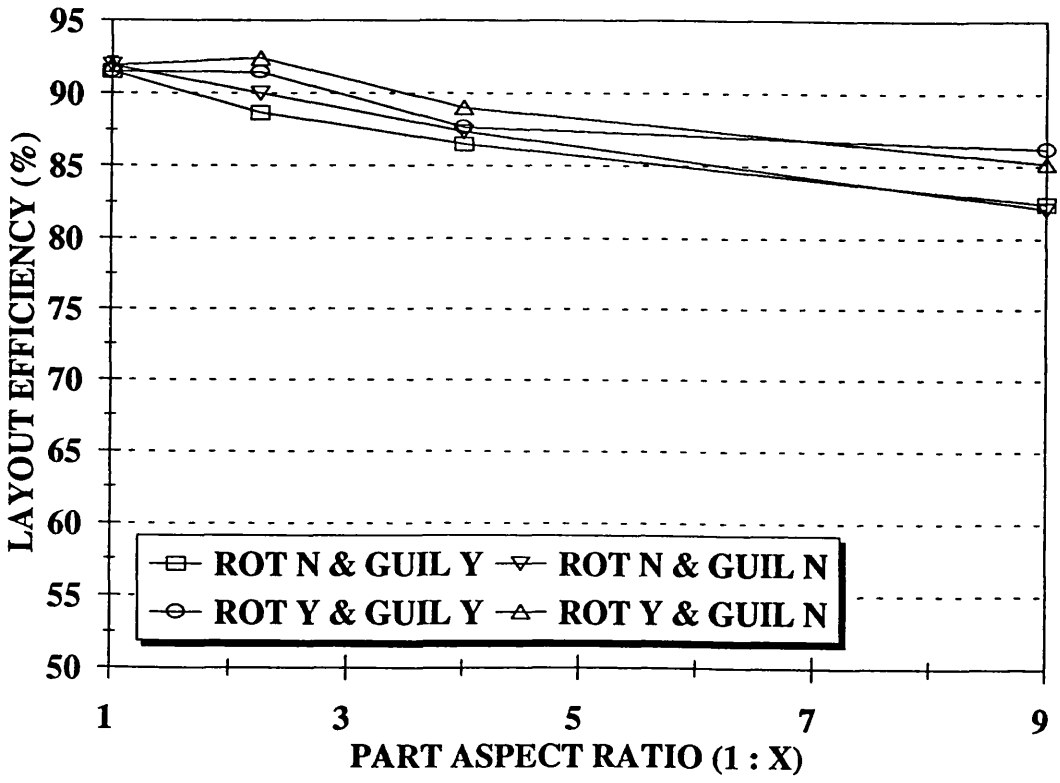


Figure 6.19.
The performance variation of the Total Fit algorithm with average part aspect ratio.

Statistical Testing of the Results

Because the layout efficiencies achieved by the algorithms do not change vastly with changes in the average part aspect ratio it needs to be established whether or not these differences in layout efficiency are significant. The results do not seem to conform to a normal distribution and therefore the Mann-Whitney test has been used to establish significance levels. To reduce testing, only the variant giving the least difference in nesting performance between the parts lists with 1:1.00 and 1:9.00 aspect ratios was tested. If this variant of the algorithm had a significant difference in layout efficiency this would also hold for all the other variants of the algorithm.

The twelve performance values for the 1:1.00 part aspect ratio list was tested against the twelve values for the 1:9.00 part aspect ratio list. The critical 'U' value to compare two sets of twelve values for a One-Tailed Mann-Whitney test at α=0.01 (99%

significance) is 31. The critical value at α=0.05 (95% significance) is 42. The two sets of data are significantly different if the 'U' value produced is less than the critical value. The values obtained are shown in Table 6.9.

ALGORITHM	PART ROT	GUIL CUT	'U' VALUE	CONFIDENCE LEVEL
NEXT FIT	Y	N	56.0	BELOW 95%
FIRST FIT	N	Y	23.0	99%
STRIP FIT	Y	N	4.0	99%
AREA FIT	Y	Y&N	49.0	BELOW 95%
TOTAL FIT	Y	N	32.5	95%
NEXT FIT	N	Y	3.0	99%
NEXT FIT	N	N	21.0	99%
NEXT FIT	Y	Y	16.0	99%
AREA FIT	N	Y&N	20.0	99%

Table 6.9. Mann-Whitney test results for effect of average part aspect ratio.

The test gives a 99% confidence level that the First Fit and Strip Fit algorithms produce a layout with less waste when nesting a list of parts with an aspect ratio of 1:1.00 rather than a list of parts with an aspect ratio of 1:9.00. The test also gives a 95% confidence level that the layouts created by the Total Fit algorithm are affected by part aspect ratio in the same manner. The best performing variants of both the Area Fit and Next Fit algorithms are robust to the effects of average part aspect ratio, giving 'U' values which do not fall within the 95% confidence level. Thus the other variants of these algorithms have been tested and all give 'U' values within the 99% confidence limit (shown in the bottom four rows of Table 6.9). Thus only two nesting constraint and algorithm combinations do not show a statistically significant difference in layout efficiency in relation to average part aspect ratio. It can be concluded that, in general terms, the layout efficiency decreases as the average part aspect ratio of the list increases.

6.5 Effect of Sheet Aspect Ratio

Purpose of the Test

If an algorithm is to place a list of parts, the aspect ratio of the sheet onto which the parts are to be placed may affect the efficiency of the layout produced. The efficiency of the layout would be generally expected to decrease with an increase in sheet aspect ratio as a narrow sheet will constrain the location of parts more than a square sheet of the same area.

Parts Lists and Sheet Sizes used in this Test

The parts lists for this test did not need to relate to each other as they all were nested on each of the sheets. Two parts lists were generated using the 'Random Method' and a third parts list using the 'Biased Aspect Ratio Method'; both methods are described in Section 4.5. The parameter values for each of these parts lists are shown in Table 6.10. Each parts list was nested onto sixteen sheets, detailed in Table 6.11. Four different sheet areas were used and for each area sheets with four different aspect ratios (1:1, 1:1.56, 1:2.78 and 1:6.25) were created. The aspect ratios were not set as integer values to allow the actual sheet dimensions to be given convenient integer values. This resulted in 48 separate test cases for each algorithm and constraint variant.

FILENAME OF PARTS LIST	NUMBER OF PARTS IN LIST	AVERAGE DIMENSION	MAXIMUM DIMENSION	AVERAGE ASPECT RATIO
SHT01	150	110mm	200mm	1:1.95
SHT02	150	183mm	340mm	1:2.12
SHT03	150	253mm	600mm	1:3.18

Table 6.10. Details of parts lists used to evaluate the effect of the sheet aspect ratio on the efficiency of the layout produced.

ASPECT RATIO 1 : 1.00	ASPECT RATIO 1 : 1.56	ASPECT RATIO 1 : 2.78	ASPECT RATIO 1 : 6.25
X = 1200mm Y = 1200mm	X = 1500mm Y = 960mm	X = 2000mm Y = 720mm	X = 3000mm Y = 480mm
X = 1500mm Y = 1500mm	X = 1875mm Y = 1200mm	X = 2500mm Y = 900mm	X = 3750mm Y = 600mm
X = 1800mm Y = 1800mm	X = 2250mm Y = 1440mm	Y = 3000mm X = 1080mm	X = 4500mm Y = 720mm
X = 2100mm Y = 2100mm	X = 2625mm Y = 1680mm	Y = 3500mm X = 1260mm	X = 5250mm Y = 840mm

Table 6.11. Details of sheet dimensions used to evaluate the effect of the sheet aspect ratio on the efficiency of the layout produced.

Results of the Test

The performance of each algorithm in relation to the sheet aspect ratio is given in Figures 6.20 to 6.24. All five graphs have been plotted in the same range to allow easy comparison. Each value given is the average of 12 individual tests, due to three parts lists being placed on four sheets of each sheet aspect ratio. The performance values for each algorithm are given in Table 6.12.

All variants of the Next Fit algorithm (Figure 6.20) initially show a slight increase in layout efficiency as the aspect ratio of the sheet increases from 1:1.00. As the sheet aspect ratio increased to more extreme values the layout efficiency decreased. The Next Fit algorithm created layouts with the majority of the waste incurred either at the ends of the strips or at the end of the sheet. The waste at the ends of the strips occupies a greater proportion of a layout consisting of strips with few member parts ie. generally this waste is inversely proportional to the sheet's Y dimension. In a multiple sheet layout the waste at the end of the sheet would increase in direct proportion to the Y dimension of the sheet. An optimum range of sheet aspect ratios can be identified in Figure 6.20, however these only apply if the sheet is orientated

with its larger dimension in the X axis. If this algorithm is used to nest a non-square sheet in its other primary orientation, in general a less efficient layout will be produced. This is because the increase in the waste at the end of the sheet will out weigh the decrease in the waste at the end of the strips. Thus the relationship shown in Figure 6.20 only holds if the larger sheet dimension is in the X axis.

ALGORITHM	ROT	GUIL	SHEET ASPECT RATIO			
			1.00	1.56	2.78	6.25
NEXT FIT	N	Y	83.3%	84.2%	83.8%	80.0%
NEXT FIT	N	N	85.3%	85.2%	85.0%	80.7%
NEXT FIT	Y	Y	84.3%	84.8%	85.0%	81.9%
NEXT FIT	Y	N	85.8%	86.4%	85.1%	81.9%
FIRST FIT	N	Y	87.5%	89.3%	91.0%	91.3%
FIRST FIT	N	N	90.7%	91.3%	93.2%	92.4%
FIRST FIT	Y	Y	88.3%	89.3%	91.8%	92.9%
FIRST FIT	Y	N	90.5%	91.1%	93.5%	93.9%
STRIP FIT	N	Y	83.0%	85.8%	87.9%	88.4%
STRIP FIT	N	N	87.3%	89.7%	91.5%	90.6%
STRIP FIT	Y	Y	87.4%	90.5%	91.9%	92.8%
STRIP FIT	Y	N	91.8%	92.4%	94.0%	94.6%
AREA FIT	N	Y&N	94.0%	94.0%	94.4%	93.3%
AREA FIT	Y	Y&N	95.8%	96.0%	96.0%	95.6%
TOTAL FIT	N	Y	91.5%	91.4%	92.2%	90.8%
TOTAL FIT	N	N	92.5%	93.2%	93.6%	92.0%
TOTAL FIT	Y	Y	91.2%	92.2%	92.4%	93.0%
TOTAL FIT	Y	N	93.6%	94.1%	93.7%	94.4%

Table 6.12. Algorithm performance values with different sheet aspect ratios.

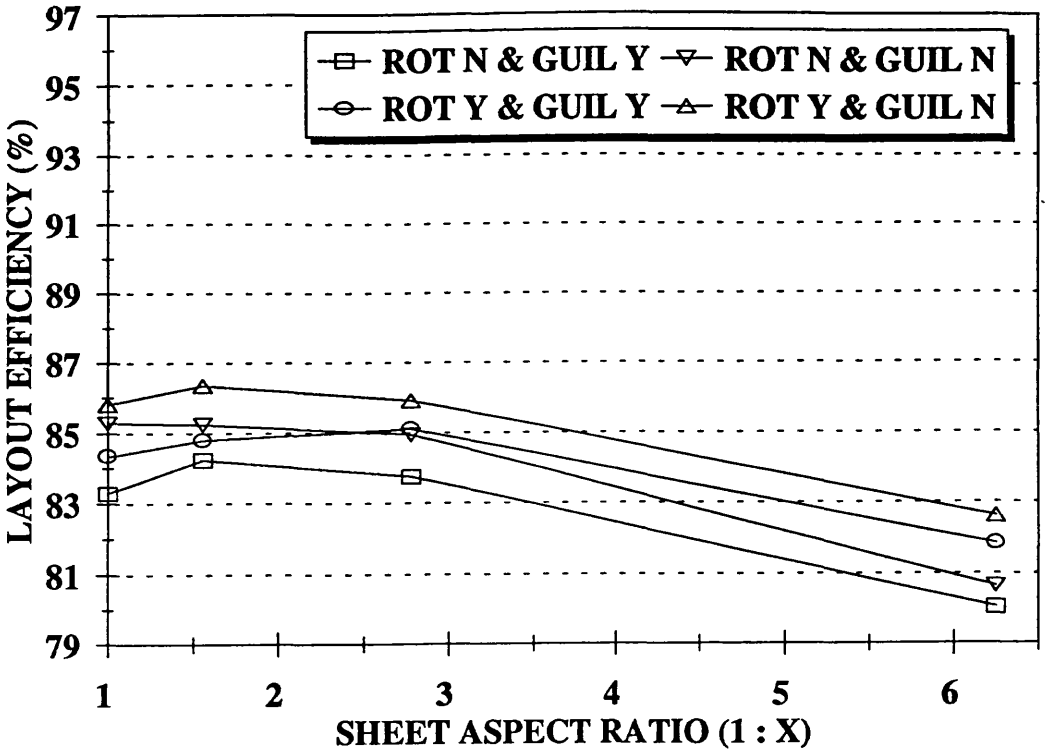


Figure 6.20.
The performance variation of the Next Fit algorithm with sheet aspect ratio.

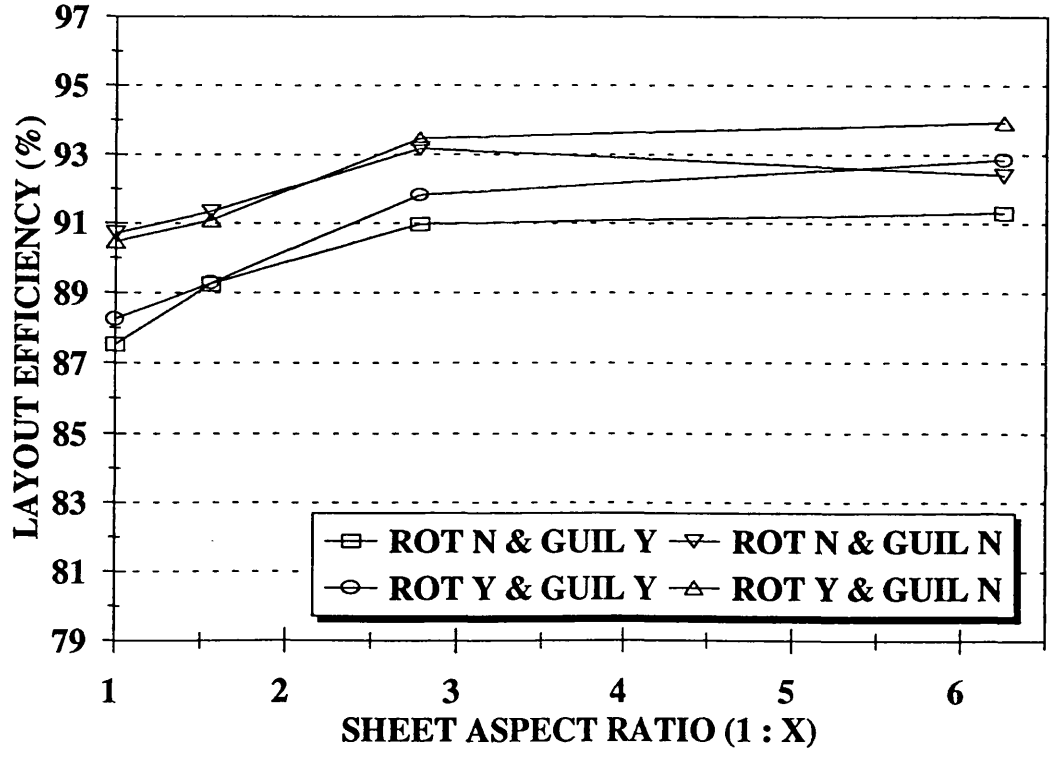


Figure 6.21.
The performance variation of the First Fit algorithm with sheet aspect ratio.

The layout efficiency of the First Fit algorithm increases as the sheet aspect ratio increases, shown in Figure 6.21. This algorithm can place parts in the 'end areas' of the existing strips which vastly reduces the waste at the end of each strip. Thus the major source of waste is that left at the end of each sheet in a multiple sheet layout, which decreases as the sheet's Y dimension decreases. As with the Next Fit algorithm this aspect ratio relationship is only true when the larger sheet dimension is placed in the X axis. If the sheet is given its other primary orientation the layout efficiency would decrease with an increase in sheet aspect ratio.

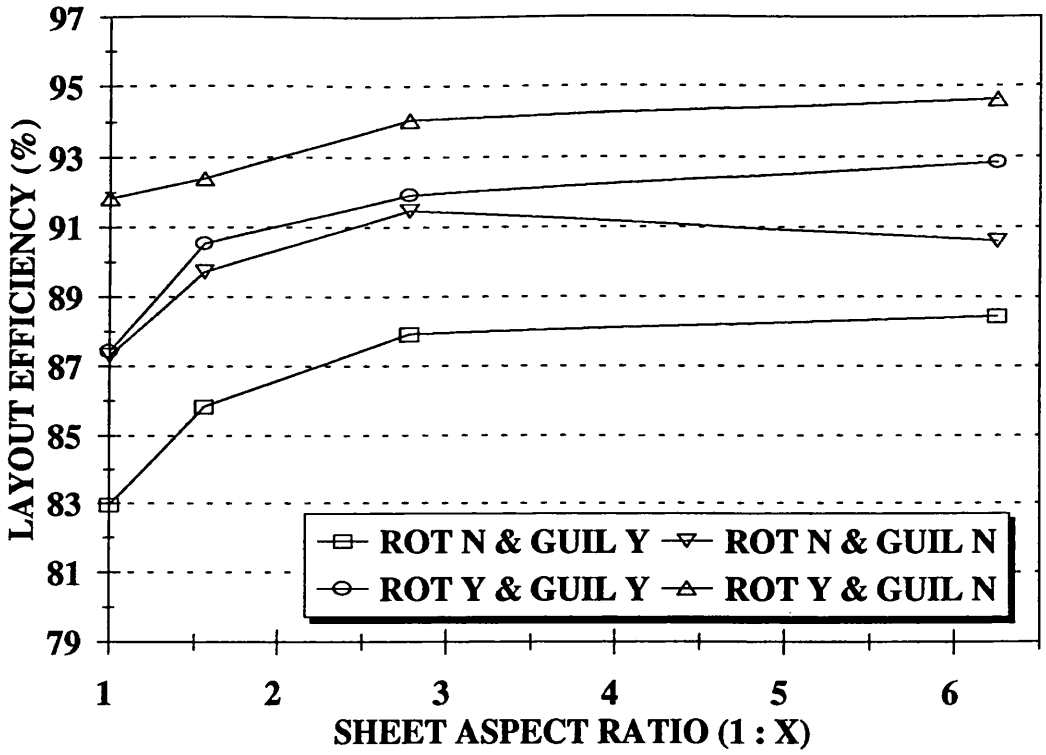


Figure 6.22.
The performance variation of the Strip Fit algorithm with sheet aspect ratio.

The layout efficiency achieved by the Strip Fit algorithm in relation to the sheet aspect ratio is shown in Figure 6.22. If the sheet has a small Y dimension less parts will be required to create a strip. Firstly, more of the sub-groups of parts with similar X dimensions will contain enough parts to form a strip, and therefore a greater proportion of the list will be placed before the strip tolerances are relaxed. Secondly, as less parts are required to form each strip more combinations will be tried; this will generally lead

to more efficient strips being created. As with the First Fit algorithm a high aspect ratio sheet is only advantageous if it is orientated with the larger dimension in the X axis.

The Area Fit algorithm, shown in Figure 6.23, shows a very slight decrease in nesting efficiency with an increase in the sheet aspect ratio when part rotation is not permitted. This is expected as, although the Area Fit algorithm is the most robust algorithm, a sheet with a high aspect ratio would still be expected to be more awkward to nest. The orientation of the sheet is irrelevant to the Area Fit algorithm as it does not operate by forming strips in a pre-determined axis.

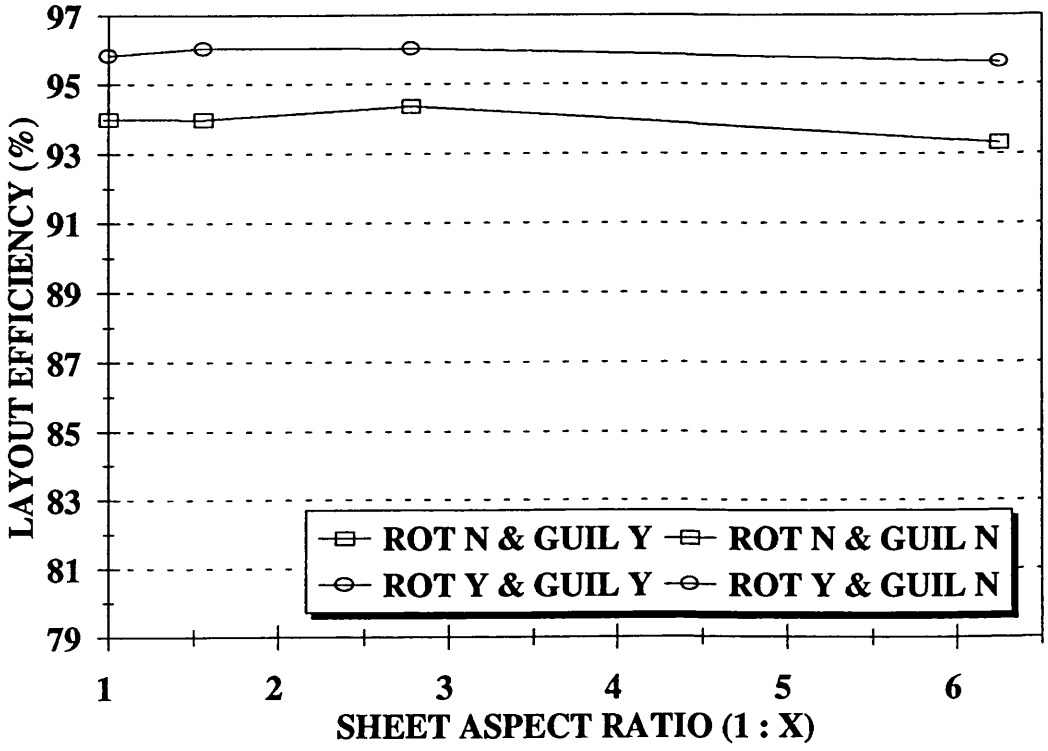


Figure 6.23.
The performance variation of the Area Fit algorithm with sheet aspect ratio.

The Total Fit algorithm combines the Strip Fit algorithm with the Area Fit algorithm. As discussed, when used alone the Strip Fit algorithm shows a significant increase in layout efficiency with sheet aspect ratio while there is a slight decrease in the efficiency of layouts created by the Area Fit algorithm. However, the performance

change in the Strip Fit algorithm is related to the relaxation of strip tolerances, which is not carried out when it is used within the Total Fit algorithm. Thus, overall, the performance of the Total Fit algorithm would be expected to be reasonably consistent across the range of sheet aspect ratios. The performance of the Total Fit algorithm with differing sheet aspect ratio is shown in Figure 6.24. Only the variant with part rotation permitted and the guillotine cut constraint imposed (ROT Y & GUIL Y) does not behave as expected. This variant shows a significant increase in layout efficiency due to part rotation permitting a greater number of parts to be placed by the Strip Fit algorithm, which creates better layouts with an increase in aspect ratio. The other variant which allows part rotation (ROT Y - GUIL N) does not behave in this manner. This is because it includes the Strip Squeeze Algorithm, which has a greater chance of improving long strips placed on a sheet with a low aspect ratio. Thus in this last variant these two effects cancel out to give a consistent performance across the range of sheet aspect ratios.

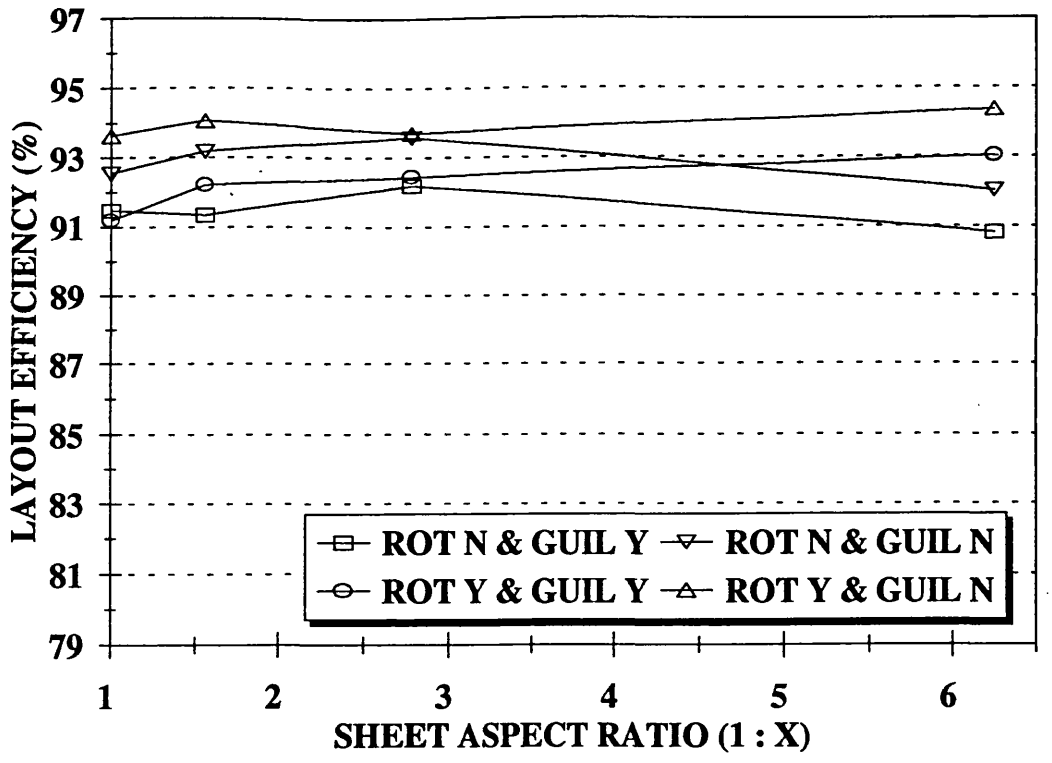


Figure 6.24.
The performance variation of the Total Fit algorithm with sheet aspect ratio.

Statistical Testing of the Results

Because the layout efficiencies achieved by the algorithms do not change vastly with changes in the sheet aspect ratio it needs to be established whether or not these differences in layout efficiency are significant. The results do not seem to conform to a normal distribution and therefore the Mann-Whitney test has been used again to establish significance levels.

As the variants of the Total Fit algorithm do not behave in the same manner in relation to the sheet aspect ratio they were all tested. For the other algorithms the variant which is least affected by sheet aspect ratio was used. If this variant of the algorithm, which has the least change in layout efficiency with changes in sheet aspect ratio, shows a significant difference in layout efficiency this should hold for all the other variants. In all cases (including the Total Fit algorithm) this was the variant with part rotation permitted and the guillotine cut constraint not applied (ROT Y - GUIL N). For each algorithm the data sets from the highest and lowest performances was tested. With the Strip Fit and Next Fit algorithms the highest results were achieved with a sheet aspect ratio of 1:6.25 and the lowest results with a sheet aspect ratio of 1:1. The highest results for the Next Fit algorithm were achieved with a sheet aspect ratio of 1:1.56 and the lowest results with a sheet aspect ratio of 1:1. The best results for the Area Fit algorithm were achieved with a sheet aspect ratio of 1:2.78 and the worst results with a sheet aspect ratio of 1:6.25.

As explained each data set contained 12 values. The critical 'U' value to compare two sets of 12 values for a One-Tailed Mann-Whitney test at α=0.01 (99% significance) is 31. The critical value at α=0.05 (95% significance) is 42. The two sets of data are significantly different if the 'U' value produced is less than the critical value. The values obtained are shown in Table 6.13.

The test gives a 99% confidence level that the Strip Fit algorithm produces a layout with less waste when nesting parts onto a sheet with an aspect ratio of 1:6.25 rather than a sheet with an aspect ratio of 1:1. The test gives a 95% confidence level that

the performance of this variant of the First Fit algorithm is affected by sheet aspect ratio. It can be assumed that this level of confidence is true for all variants of these two algorithms. The variant of the Next Fit algorithm tested gives a 'U' value which was below the 95% confidence threshold. Thus the other variants of this algorithm were also tested, the results of which are shown in the lower section of Table 6.13 and all give 'U' values outside the threshold for the 95% confidence level. The variant of the Area Fit algorithm which was tested gives a 'U' value below the 95% confidence threshold, requiring the other variant of this algorithm to also be tested. The other Area Fit variant, shown in the lower section of Table 6.13, also has a 'U' value which falls outside the 95% confidence level. All but one of the Total Fit algorithm's variants also have 'U' values which fall outside the 95% confidence level. It can be concluded that, although the layout efficiency is affected by the sheet aspect ratio, the effect is not as significant as with the other features tested.

ALGORITHM	PART ROT	GUIL CUT	'U' VALUE	CONFIDENCE LEVEL
NEXT FIT	Y	N	53.0	BELOW 95%
FIRST FIT	Y	N	41.0	95%
STRIP FIT	Y	N	22.0	99%
AREA FIT	N	Y&N	72.0	BELOW 95%
TOTAL FIT	N	Y	43.5	BELOW 95%
TOTAL FIT	N	N	46.0	BELOW 95%
TOTAL FIT	Y	Y	34.0	95%
TOTAL FIT	Y	N	53.5	BELOW 95%
NEXT FIT	N	Y	48.0	BELOW 95%
NEXT FIT	N	N	47.5	BELOW 95%
NEXT FIT	Y	Y	54.0	BELOW 95%
AREA FIT	Y	Y&N	59.5	BELOW 95%

Table 6.13. Mann-Whitney test results for effect of sheet aspect ratio.

6.6 Effect of Sheet Area to Average Part Area Ratio

Purpose of the Test

If an algorithm is to place a list of parts, the size of the parts in relation to the sheet onto which they are to be placed may affect the efficiency of the layout produced. The ratio of sheet area to average part area is equivalent to the number of parts which would fit on the sheet if 100% nesting efficiency were achieved. The efficiency of the layout would be expected to generally fall with a decrease in this ratio. This is because when relatively large parts are nested on a sheet there will be many reasonably large spaces around the placed parts which are still too small for any of the remaining parts to occupy (a good example of this is shown in Figures 5.1 and 5.5 where the same parts list has been nested by the Next Fit algorithm onto two different sized sheets). The sheet area to average part area ratio would be expected to affect all the algorithms to some extent.

Parts Lists and Sheet Sizes used in this Test

In this test it was necessary to create sets of sheets and parts lists which had a known ratio of sheet area to average part area. This was most easily achieved by applying the same parts list to a series of different sized sheets. To maximise the range of the test, while keeping the number of sheets low, each subsequent sheet was made double the area of its predecessor. As six sheet sizes were created in this manner, the test covers a 32 fold increase in the sheet size. Thus, when one parts list is nested onto all the sheets within such a set, the ranges of 'sheet area to average part area ratio' will also cover a 32 fold increase. Unfortunately, if the sheet aspect ratios are to be held at the same value, this cannot be achieved with integer sheet dimensions. However, a series of sheet sizes were found in which the aspect ratio's alternate as described below and in Table 6.14.

In Section 6.5 sheet aspect ratio was shown to affect nesting efficiency and the aim of this test was to isolate the effect of the sheet area to average part area ratio on layout efficiency. Although, the test required a number of sheet and parts list combinations to be tested, it was possible to fix the average sheet aspect ratio to a common value for each sheet area to average part area ratio.

This is most easily done by creating pairs of sheet sets, with the aspect ratios of the sheets in the first set altenating in reverse sequence to those of the second set. Four sets of six sheets (A1..A6, B1..B6, C1..C6 & D1..D6) were used in this test (Table 6.14). The numbers denote the sheet area to average part area ratio, the actual values of which will be dicussed later. The 'A' and 'B' sets' aspect ratios, by alternating in an opposite sequence, give a common average sheet aspect ratio for all sheet sizes. The aspect ratios of the 'C' and 'D' sets also alternate in an opposite sequence.

In the nesting system developed, the sheet is automatically allocated a perimeter bridge gap (this is achieved by removing the bridge gap from the dimensions of the sheet area available to nest). To maintain an exact doubling of the area available to nest throughout the series of sheets, the bridge gap was added to both the X and Y dimensions of all the sheets.

Two parts lists were placed on the four member sheets of each set (A to D), ie. 8 parts lists in total. Details of these parts lists are given in Table 6.15 along with the sheet set to which they were applied. As each sheet set contains 6 sheets there are 48 test cases for each algorithm and constraint variant. The parts lists have been created to give pre-determined sheet area to average part area ratios for each of their dedicated set members. The ratios for the set members (1 to 6) are 11.43, 22.86, 45.71, 91.43, 182.86 and 365.71. The average aspect ratio of the parts does not need to be fixed at one value as each list is nested once on each sheet within the set. There were 150 parts in each list to ensure that 40-50% of the largest sheet area was occupied. In practice up to 20 of the smallest sheets were required to nest the parts list.

SET	SHEET DIMENSIONS	SHEET AREA TO NEST	ASPECT RATIO
A1	610mm by 410mm	240000mm^2	1:1.5
A2	810mm by 610mm	480000mm^2	1:1.3
A3	1210mm by 810mm	960000mm^2	1:1.5
A4	1610mm by 1210mm	1920000mm^2	1:1.3
A5	2410mm by 1610mm	3860000mm^2	1:1.5
A6	3210mm by 2410mm	7720000mm^2	1:1.3
B1	810mm by 610mm	480000mm^2	1:1.3
B2	1210mm by 810mm	960000mm^2	1:1.5
B3	1610mm by 1210mm	1920000mm^2	1:1.3
B4	2410mm by 1610mm	3860000mm^2	1:1.5
B5	3210mm by 2410mm	7720000mm^2	1:1.3
B6	4810mm by 3210mm	15440000mm^2	1:1.5
C1	760mm by 410mm	300000mm^2	1:1.7
C2	1010mm by 610mm	600000mm^2	1:1.9
C3	1510mm by 810mm	1200000mm^2	1:1.7
C4	2010mm by 1210mm	2400000mm^2	1:1.9
C5	3010mm by 1610mm	4800000mm^2	1:1.7
C6	4010mm by 2410mm	9600000mm^2	1:1.9
D1	1010mm by 610mm	600000mm^2	1:1.9
D2	1510mm by 810mm	1200000mm^2	1:1.7
D3	2010mm by 1210mm	2400000mm^2	1:1.9
D4	3010mm by 1610mm	4800000mm^2	1:1.7
D5	4010mm by 2410mm	9600000mm^2	1:1.9
D6	6010mm by 3210mm	19200000mm^2	1:1.7

Table 6.14. The sheet sets used to test the sheet area to average part area ratio.

The 'Fixed Area Method', described in Section 4.5, was used to set the average part area at the exact value required. This method operates by repeatedly splitting a given rectangular area into groups of 10 parts to create the list size required (held to multiples of 10). The parts generated by this method are then reduced in each dimension by the bridge gap. Thus the sheet area to average part area ratio accounts for the bridge gap and thus accurately reflects the number of parts which would be placed on the sheet with a 100% efficient layout. To allow the rectangular area (to be split into sets of 10 parts) to be given convenient integer dimensions, the resulting sheet area to average part area ratio's do not have integer values.

SHEET SET	FILENAME OF PARTS LIST	NUMBER OF PARTS	AVERAGE DIMENSION	MAXIMUM DIMENSION	AVERAGE ASPECT RATIO
A	RAT01	150	135mm	250mm	1:1.98
A	RAT02	150	145mm	362mm	1:2.81
B	RAT03	150	195mm	350mm	1:1.77
B	RAT04	150	201mm	451mm	1:2.41
C	RAT05	150	152mm	300mm	1:2.03
C	RAT06	150	153mm	325mm	1:2.09
D	RAT07	150	221mm	424mm	1:1.72
D	RAT08	150	227mm	544mm	1:2.77

Table 6.15. The parts lists used to evaluate the effect of the sheet area to average part area ratio.

Results of the Test

The performance of each algorithm in relation to the sheet area to average part area ratio is given in Figures 6.25 to 6.29. All five graphs have been plotted in the same range to allow easy comparison. Each value given is the average of eight test cases, corresponding to the eight parts lists nested. The performance values for each algorithm are given in Table 6.17.

ALGO	ROT	GUIL	SHEET AREA TO AVERAGE PART AREA RATIO					
			11.4	22.9	45.7	91.4	182.9	365.7
NEXT	N	Y	65.6%	76.1%	80.5%	84.6%	86.4%	87.8%
NEXT	N	N	66.1%	76.3%	82.5%	88.6%	92.2%	92.9%
NEXT	Y	Y	65.6%	77.8%	80.7%	85.0%	88.9%	89.6%
NEXT	Y	N	65.9%	78.1%	83.8%	87.3%	92.4%	94.3%
FIRST	N	Y	78.2%	85.3%	88.9%	90.2%	91.3%	90.6%
FIRST	N	N	78.2%	85.8%	89.7%	93.9%	95.9%	96.1%
FIRST	Y	Y	75.0%	83.1%	87.4%	88.6%	91.7%	91.3%
FIRST	Y	N	75.2%	83.2%	89.8%	92.4%	96.0%	96.1%
STRIP	N	Y	84.5%	86.4%	87.5%	86.4%	80.5%	78.9%
STRIP	N	N	84.7%	89.1%	89.4%	90.4%	88.8%	86.3%
STRIP	Y	Y	87.9%	89.4%	91.2%	89.8%	88.3%	84.8%
STRIP	Y	N	88.8%	90.3%	93.1%	93.5%	93.4%	90.3%
AREA	N		91.9%	93.2%	93.9%	93.9%	93.1%	90.4%
AREA	Y		94.4%	95.2%	95.5%	95.3%	95.1%	92.6%
TOTAL	N	Y	88.3%	90.3%	92.1%	92.2%	92.8%	90.8%
TOTAL	N	N	88.2%	91.6%	93.5%	93.7%	94.4%	92.5%
TOTAL	Y	Y	88.2%	90.5%	91.3%	92.2%	92.7%	89.1%
TOTAL	Y	N	88.9%	90.9%	92.7%	94.5%	96.3%	92.4%

Table 6.17. The variation in algorithm's performance with sheet area to average part area ratio.

As expected all variants of the Next Fit algorithm, Figure 6.25, show a marked improvement in nesting efficiency as the ratio of sheet area to average part area increases. Parts which are small relative to the sheet form strips which, on average, have less end waste. In addition the strip widths are small relative to the sheet width resulting in less waste at the end of the sheet. The improvement in layout efficiency achieved by the Strip Squeeze algorithm also increases with the sheet area to average part area ratio. Parts which are small relative to the sheet form relatively thinner strips with a considerable taper, which are ideal for improvement by the Strip Squeeze algorithm.

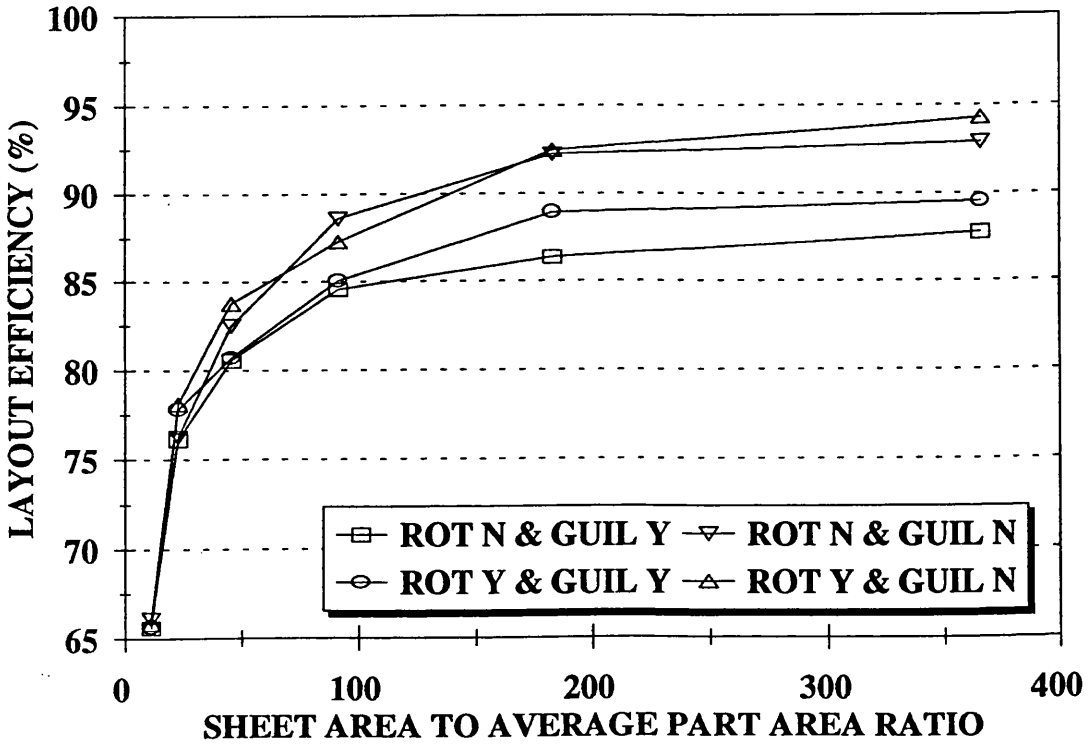


Figure 6.25.
The performance variation of the Next Fit algorithm with the sheet area to average part area ratio.

The First Fit algorithm, Figure 6.26, performs in a similar manner to the Next Fit algorithm. However, the performance at low sheet area to part area ratios is significantly better due to the ability of this algorithm to place parts in the strip 'end areas'. A clear example of this difference in the operation of the First Fit and Next

Fit algorithms is shown in Figures 5.10 and 5.11 (Section 5.3). The benefit of this method of placement is reduced with higher ratio values, as the strip 'end areas' are proportionally smaller.

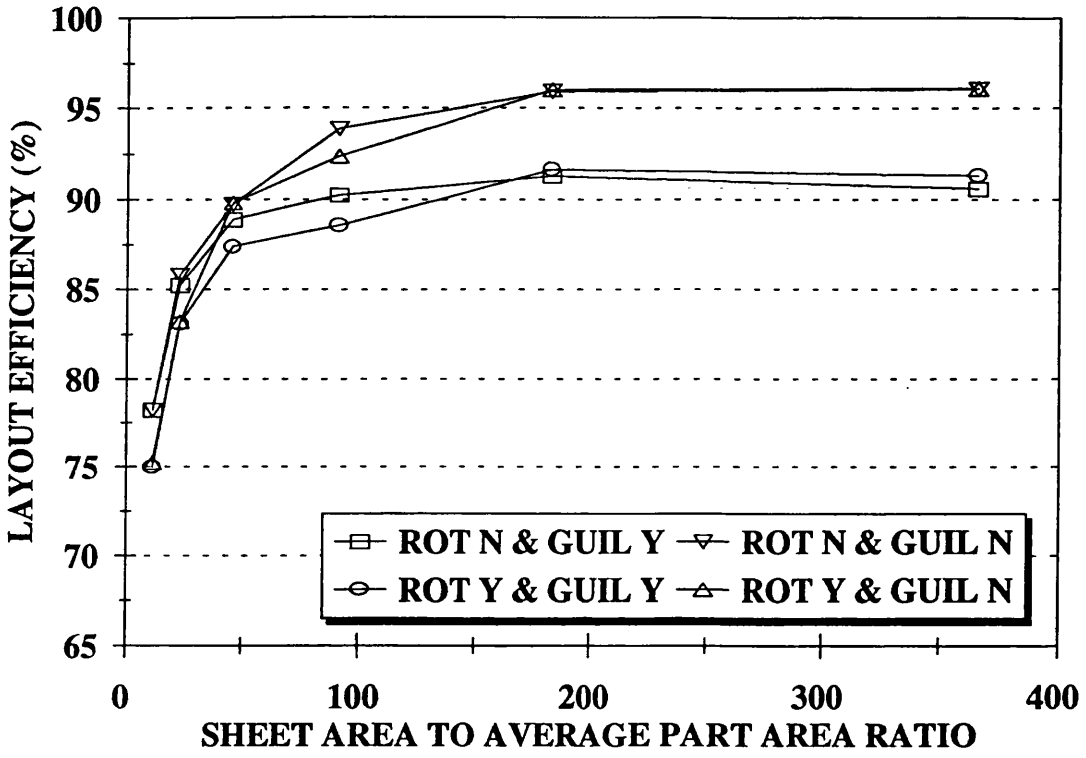


Figure 6.26.
The performance variation of the First Fit algorithm with the sheet area to average part area ratio.

The performance of the Strip Fit algorithm in relation to changes in the sheet area to part area ratio is shown in Figure 6.27. All the variants show an initial increase in layout efficiency, as initially efficient strip combinations of such large parts, in comparison to the sheet, are difficult to find. Each algorithm variant then peaks at the level where the constraints imposed allow a large number of strips to be formed without tolerance relaxation. Allowing part rotation will, on average, almost double the number of parts found with a similar width which can be placed as a strip. This will cause the algorithm to achieve its most efficient layouts on a 'wider' sheet. Thus the peak in layout efficiency for variants with part rotation (ROT Y) will be at a hgher ratio value than those without. Also as the ratio increases the gain made in

layout efficiency by using the Strip Squeeze algorithm will increase. This will cause the variants with the Strip Squeeze algorithm applied (GUIL N) to peak at a higher ratio value than the variants without the Strip Squeeze algorithm applied. Thus the variant with both constraints applied (ROT N - GUIL Y) peaks at the second lowest ratio value tested (22.86). If part rotation is permitted the peak moves to the third ratio value tested (45.71). The performance of both variants with the guillotine cut constraint not applied (GUIL N) peak at the fourth ratio value tested (91.43). In addition, the less constrained the algorithm the more consistent its performance over the range of the test.

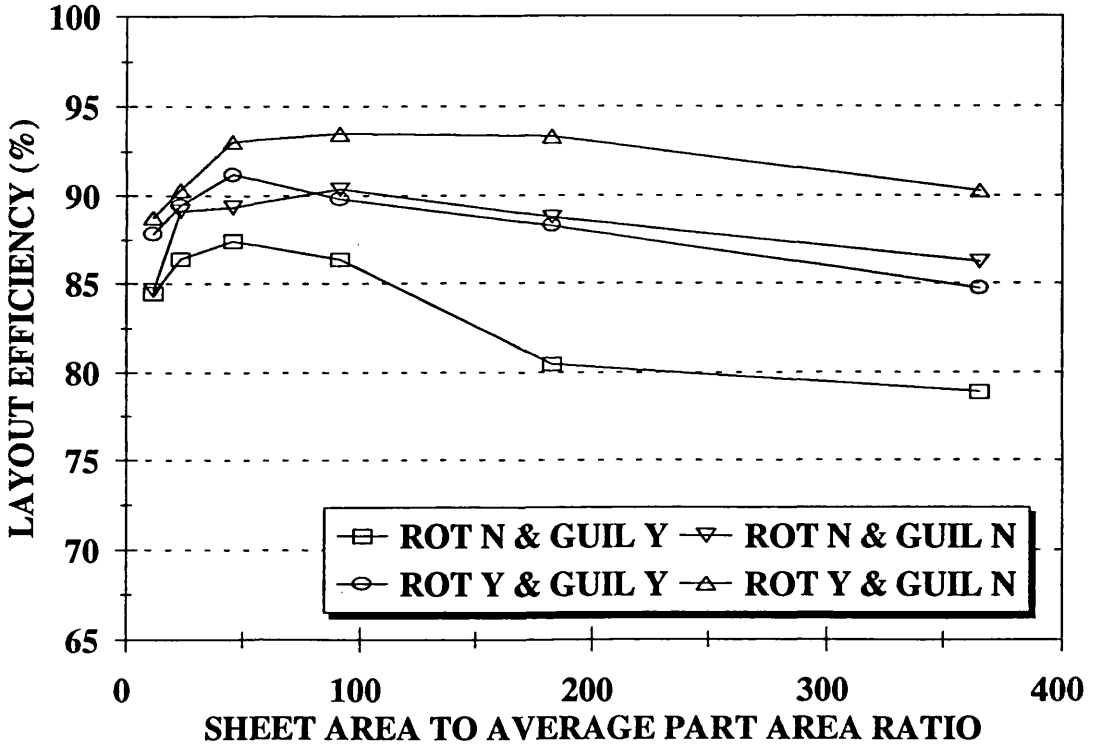


Figure 6.27.
The performance variation of the Strip Fit algorithm with the sheet area to average part area ratio.

The performance of the Area Fit algorithm in relation to changes in the sheet area to part area ratio is shown in Figure 6.28. Both variants behave in a similar manner throughout the test range, although allowing the rotation of the parts gives a consistent layout efficiency improvement of approximately 2.5%. The algorithm shows an initial

increase in layout efficiency due to the first two sets of lists containing parts which are sufficiently large, relative to the sheet, to be awkward to place. The performances of the variants peak at the third (45.71) and fourth (91.45) ratio value tested. The performance drops significantly with the largest sheet area to part area ratio (365.71). This is because this situation is better suited to the 'dimension fit' and 'bottom left' part placement methods (see Section 4.7) rather than the 'perfect fit' method. The algorithm could be adapted to nest more efficiently for this extreme sheet size by tightening the 'dimension fit' tolerance, which is currently set at 90%. This algorithm is far more robust to changes in the sheet area to part area ratio than the other algorithms tested.

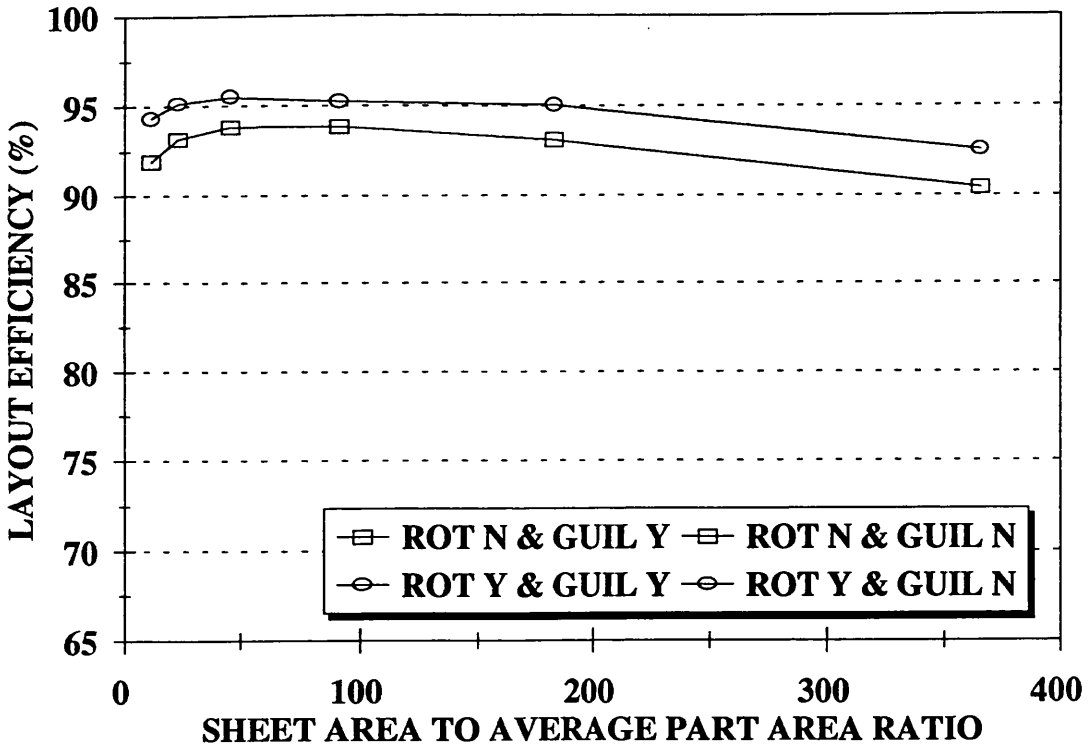


Figure 6.28.
The performance variation of the Area Fit algorithm with the sheet area to average part area ratio.

The Total Fit algorithm, Figure 6.29, has the same curve shape as the Area Fit and Strip Fit algorithms. This is expected as the Total Fit algorithm is a combination of these two algorithms. All variants of the Total Fit algorithm peak at the fifth test point, which has a sheet area to part area ratio of 182.86. However this peak, as with

the last two algorithms, could be moved by setting the algorithms acceptance tolerances at different values. With a high sheet area to average part area ratio it is considered that the layout efficiency would be improved by relaxing the strip width tolerance of the Strip Fit algorithm in this application. The resulting layout improvement would be even greater if the Strip Squeeze algorithm was then also applied.

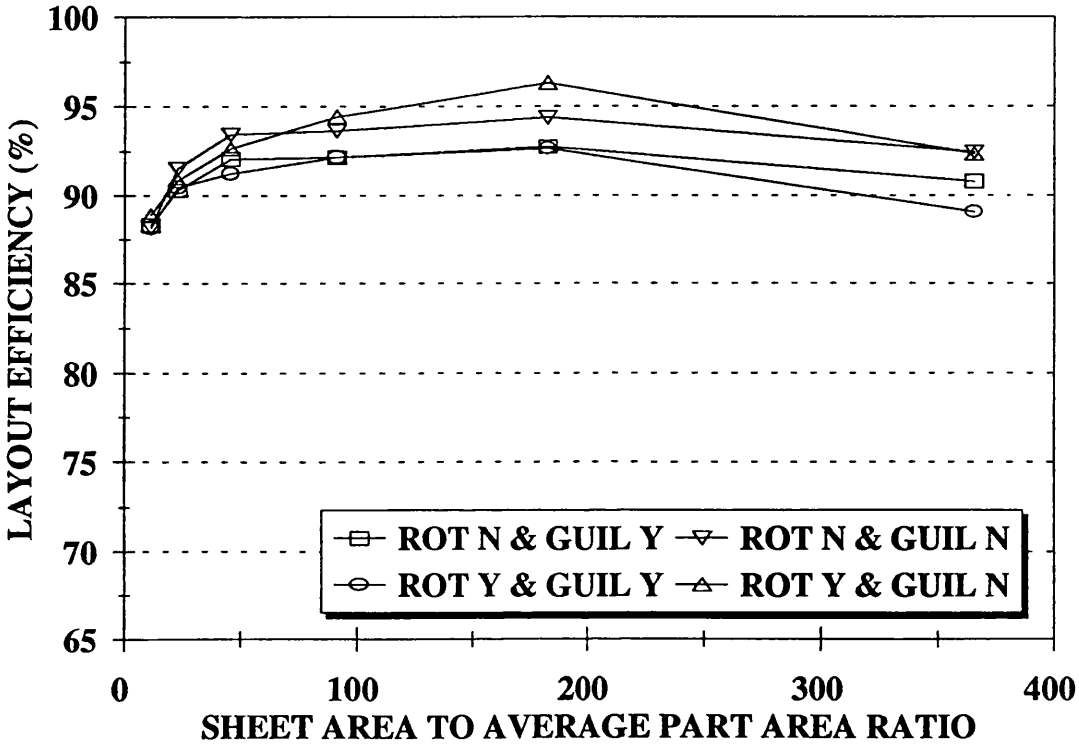


Figure 6.29.
The performance variation of the Total Fit algorithm with the sheet area to average part area ratio.

Statistical Testing of the Results

Because the layout efficiencies achieved by the algorithms do not change vastly with changes in the sheet area to average part area ratio it was necessary to established whether or not these differences in layout efficiency are significant. The results do not seem to conform to a normal distribution and therefore the Mann-Whitney test has again been used to establish significance levels. To reduce the testing, only the variant with the most consistent performance was tested. For all the algorithms this is the variant with part rotation not permitted and the guillotine cut constraint imposed (ROT N - GUIL Y).

There are eight performance values for each parts list size. The critical 'U' value to compare two sets of eight values for a One-Tailed Mann-Whitney test at α=0.01 (99% significance) is 10. The critical value at α=0.10 is 15 (95% significance). The two sets of data are significantly different if the 'U' value produced is less than the critical value. The values obtained are shown in the upper section of Table 6.18.

The performance of the three new algorithms peaks in the middle of the ratio range (see Figures 6.27, 6.28 & 6.29). Thus two significance tests were carried out for each; a value comparing the performance at the maximum ratio value with the peak performance ratio value (denoted 'MAX & PEAK' in Table 6.17) and a value comparing the performance at the minimum ratio value with the peak performance ratio value (MIN & PEAK). If a significant difference exists in the performance of the most consistent variant of the algorithm this should hold for the other variants.

There is 99% confidence that all the algorithms are significantly affected in some manner by the ratio of sheet area to average part area. The only test to give a 'U' value below the 95% confidence threshold is the comparison between the minimum and peak performance ratio values (MIN & PEAK) for the Strip Fit algorithm. This required the other three variants of this algorithm to be tested. The resulting 'U' values are shown in the lower section of Table 6.18 and all show a significant difference in the performance at the two ratio values tested.

ALGORITHM	PART ROT	GUIL CUT	'U' VALUE	CONFIDENCE LEVEL
NEXT FIT	N	Y	0.0	99%
FIRST FIT	N	Y	0.0	99%
STRIP FIT (MIN & PEAK)	N	Y	17.0	BELOW 95%
STRIP FIT (MAX & PEAK)	N	Y	0.0	99%
AREA FIT (MIN & PEAK)	N	Y	2.5	99%
AREA FIT (MAX & PEAK)	N	Y	0.0	99%
TOTAL FIT (MIN & PEAK)	N	Y	0.0	99%
TOTAL FIT (MAX & PEAK)	N	Y	1.0	99%
STRIP FIT (MIN & PEAK)	N	N	8.0	99%
STRIP FIT (MIN & PEAK)	Y	Y	12.5	95%
STRIP FIT (MIN & PEAK)	Y	N	3.0	99%

Table 6.18. Mann-Whitney test results for effect of sheet area to average part area ratio.

In summary, the efficiencies of the layouts produced by the First Fit and Next Fit algorithms increase with an increase in this ratio. The efficiencies of the layouts produced by the Strip Fit, Area Fit and Total Fit algorithms each peak at different values of the ratio. The performance of all three algorithms either side of this peak deteriorates significantly.

6.7 Algorithm Run Times

All testing has been carried out on a PC with an Intel 386 chip and a 33MHz clock speed. The Area Fit algorithm required the longest run time of all the algorithms when nesting the 160 parts of the SIZ35 list onto two 3000mm by 2000mm sheets. The creation of this layout required slightly less than eight seconds. The Area Fit algorithm is invariably the slowest of all the algorithms tested. On average the Strip Fit and Total Fit algorithms require just over half the time of the Area Fit algorithm. The two benchmark algorithms are radically faster, never requiring more than one second to place even the largest list.

As all the algorithms run so fast, the processing time is not considered to be an important factor in the evaluation of their overall performance. Even the slowest algorithm operates far faster than would be required in a practical situation. This is useful, however, as it allows more time to be applied to efficiently placing parts within the rectangular envelopes prior to nesting. It also makes the further development of the sophistication of the algorithms a practical possibility. Another option is to run more than one of the algorithms and select the most efficient layout generated.

Chapter 7

Conclusions

In Chapter 1 a general review has been carried out of the planning of sheet metal cutting and the general methods and problems involved in the creation of layouts of parts (nesting). This has included the development of a computer aided or fully automated nesting system and its required data outputs. A comprehensive review of automated two dimensional nesting systems has been carried out as well as other areas which may use methods which are applicable to this problem (Chapter 2). These include one and three dimensional packing problems, the facilities layout problem and methods of part simplification.

An overall plan for a new nesting system has been described in Chapter 3. This nesting system operates in two stages; initially individual or groups of parts are enclosed in rectangular envelopes and these are subsequently 'nested' onto the required sheet size by a placement algorithm. Proposed methods of enclosing clusters of parts in rectangular envelopes and of creating tessellating patterns of identical parts within rectangular envelopes are also described in this chapter. However, as these methods have not been implemented and tested, they have not been evaluated quantitatively.

The core of the nesting system is the placement of the rectangular envelopes onto stock sheets. Two new methods of placing rectangular envelopes, the Strip Fit algorithm and the Area Fit algorithm, are introduced. These can be used individually or together in series, as is the case with the Total Fit algorithm. A third new algorithm, the Strip Squeezing algorithm, has also been developed which can be used

to improve any layout which consists of strips of parts. The basic operation of these algorithms is described in Sections 3.6 to 3.8. The details of how these algorithms have been implemented into a nesting system is given in Chapter 4. To provide a benchmark two established algorithms, the Next Fit and the First Fit algorithms [Coffman et al (1980)], have also been included. Both of these algorithms have been used with and without the new Strip Squeeze algorithm.

A qualitative evaluation of the algorithms can be made from the layouts shown in Chapter 5. The algorithms implemented have been rigorously tested across a broad range of nesting situations, which are described in Chapter 6, allowing a quantitative evaluation of their performances to be carried out. All of the algorithms operated sufficiently quickly (maximum 8 seconds), compared to the total planning time in industry, for the processing times to be irrelevant to the evaluation of the best layout.

Of the algorithms tested, the Area Fit algorithm has been shown to give the best overall performance in terms of the highest mean layout efficiency and the greatest number of 'best layouts'. This is due to poor acceptance tolerances for the Total Fit algorithm being chosen during initial testing, as it contains the Area Fit algorithm. Better acceptance tolerances should allow the Total Fit algorithm to out perform the Area Fit algorithm. The Total Fit algorithm gives the next best performance with an overall mean layout efficiency which is 1.45% below that of the Area Fit algorithm. Also, the Total Fit algorithm's layout efficiencies have a smaller standard deviation and therefore its performance is more consistent than that of the Area Fit algorithm.

The remaining algorithms are ranked from best to worst as First Fit, Strip Fit and Next Fit. The First Fit and Strip Fit algorithms produce a reasonable number of 'best layouts' and would be worth using if a number of layouts were to be created and the best used. This is particularly true when no constraints apply to the Strip Fit algorithm or when the guillotine cut constraint is not imposed on the First Fit algorithm, allowing the Strip Squeeze algorithm to be used to improve the layout.

Where applicable the Strip Squeeze algorithm gives an average layout efficiency

improvement of around 2%. The Strip Fit algorithm is not intended to operate alone and therefore would not be expected to out perform the Area Fit, Total Fit or First Fit algorithms. Accepting this, the new placement algorithms which are intended to operate alone (Area Fit and Total Fit) have out performed the benchmark algorithms and therefore are an advancement in the field of automated nesting.

In addition to the direct evaluation of the algorithms performance, the effect of four nesting parameters were also tested. These were the number of parts in the list to nest, the average aspect ratio of the parts, the aspect ratio of the sheets onto which the parts are to be placed and the ratio of the sheet area to the average part area. The last parameter is effectively the size of the parts relative to the sheet.

The number of parts to nest had a very marked effect on the performance of the Strip Fit algorithm and a statistically significant effect on all the other algorithms except the Next Fit algorithm. The performance of all of the algorithms increased with an increase in the number of parts in the list.

The performance of all the algorithms decreased with an increase in average part aspect ratio. Again the most extreme effect is on the Strip Fit algorithm. The performances of the Area Fit and Total Fit algorithms are affected to a far lesser extent than the other algorithms.

The sheet aspect ratio has the least effect of all the parameters and again the Area Fit and Total Fit algorithms are least affected by the changes, with a reasonably consistent performance. The performance of the First Fit and Strip Fit algorithms increases with sheet aspect ratio, which is opposite to the expected effect. The performance of the Next Fit algorithm peaks between a sheet aspect ratio of 1:1.00 and 1:2.50. For all but the Area Fit algorithm, this relationship only holds if the largest sheet dimension is placed in the X axis.

Finally the sheet area to part area ratio has a significant effect on the performances of all the algorithms. The performance of the First Fit and Next Fit algorithms

increases with an increase in the ratio, however the performances of the three new algorithms peak within the range of ratios tested. Changing the acceptance tolerances within these algorithms would alter the ratio value at which the performance peaks.

Of the nesting problem parameters, the average part aspect ratio and the sheet aspect ratios have a far less significant effect on layout efficiency than the size of the list of parts and the ratio of sheet area to average part area (effectively sheet size).

From this it can be concluded that the Area Fit and Total Fit algorithms give the best mean performance and are the most robust to awkward nesting situations of the algorithms tested.

The Strip Squeezing algorithm has been shown to give a significant improvement in layout efficiency for all of the algorithms tested which form strips of parts.

Chapter 8

Further Work

The current nesting system could be developed further by improving the performance of the rectangular envelope nesting algorithms or by increasing the system's scope. The latter would require a system to extract the necessary part data from a CAD source and a system to enclose the parts within rectangular envelopes.

Improving the rectangular envelope nesting performance.

The Strip Squeeze algorithm is an exact calculative method, thus having no acceptance tolerances for which better values may be found, and its use is held to specific points in the host algorithm's operation. Therefore no further development of the Strip Squeeze algorithm is foreseen.

The Area Fit and Strip Fit algorithms both contain a number of acceptance tolerances for which good values were found by trial and error. It is likely that optimum values of these tolerances could be found by methodical experimentation. This could also lead to the identification of optimum tolerance values to match different features of the parts list to be nested. An expert system could be added to this nesting system which examined the parts list to be placed and set the best algorithm tolerances for the situation. This approach could be expanded to recognise when a particular algorithm would be most appropriate to use. The First Fit algorithm could be improved by adding a method of nesting parts into the area remaining on a sheet when the next part in the list is too large to place. This could be achieved by a method similar to that

employed by the Strip Fit algorithm or by nesting this area in the same way as the area at the end of each strip. There may also be better sequences of part placement to use in particular nesting situations.

As no algorithm required more than eight seconds to create a layout of 160 parts, it would be viable to create a number of layouts and choose the best. The benefit of this approach is highlighted by two features of the test results. Although the Area Fit algorithm is clearly the best overall of those tested, it did not always give the best layout for a specific case and occasionally gave a significantly poorer layout than one of the other algorithms. Secondly, even the algorithms which generally performed poorly would occasionally produced the best layout. This is due to the particular parts, by chance, fitting the sheet well when placed by such an algorithm. Thus a system which tries a number of algorithms and chooses the best layout produced would be highly effective.

Qu and Sanders 1989 describe a method of optimising the stock sheet sizes which are used for the nesting of a particular list of parts (Section 2.7). This method could be incorporated into the system and, in addition, each sheet could be tried in both primary orientations. Thus a 'greedy heuristic' approach could be used which would optimise the size and orientation of each sheet in the layout before starting a new sheet. This could be repeated for a small number of algorithms which are known to give good results.

Expanding the scope of the system.

The system is currently limited to producing layouts of rectangles which represent the rectangular envelopes placed around actual parts. Sections 3.3 to 3.5 specify methods to place the parts within rectangular envelopes either individually, in clusters of different parts or in a tessellating layout. A system could be developed to import parts from a specified CAD source and then place the parts within rectangular envelopes for subsequent placement by the algorithms developed and tested in this work. This

would allow the system to be evaluated against 'non-rectangular' nesting algorithms and the systems which are currently applied to the problem in industry. The full automation of this system could be completed by passing the finished layout to a CAPP system to plan the path of the cutting tool and generate the required NC code or operator instructions. The operation and requirements of these systems are discussed in Section 1.7.

References.

Adamowicz M. and Albano A. (1976a). Nesting two-dimensional shapes in rectangular modules. *Computer Aided Design.* Vol. 8, No. 1, January, pp 27-33.

Adamowicz M. and Albano A. (1976b). A solution of the rectangular cutting-stock problem. *IEEE Trans. Syst. Man. And Cybernetics.* Vol. SMC-6, No. 4, April, pp 303-310.

Albano A. and Sapuppo G. (1980). Optimal Allocation of Two-Dimensional Irregular Shapes Using Heuristic Search Methods. *IEEE Trans. on Systems, Man and Cybernetics.* Vol SMC-10, No 5, May, pp 242-248.

Albano A. and Orsini R. (1979a). A tree search approach to the M-partition and Knapsac problems. *The Computer Journal.* Vol 23, No 3, pp 256-261.

Albano A. and Orsini R. (1979b). A heuristic solution of the rectangular cutting stock problem. *The Computer Journal.* Vol 23, No 4, pp 338-343.

Albano A. (1977). A method to improve two-dimensional layout. *Computer-aided Design.* Vol 9, No 1, January, pp 48-52.

Alting L. and Zhang H. (1989). Computer Aided Process Planning: the state-of-the-art survey. *INT. J. PROD. RES.* Vol 27, No 4, pp 553-583.

Arndt B. (1979). Experiences with SMD - A CAM-Program Package on Minicomputers. *Compt. App. in the Auto. of Shipyard Operation and Ship Design III.* Eds. Kuo C. MacCallum K. J. & Williams T.J. IFIP, N. Holland Pub. Co. pp 163-169.

Baker B.S., Coffman E.G.Jr. and Rivest R.L. (1980). Orthogonal packing in two dimensions. *SIAM J. COMPUT.* Vol 9, No 4, Nov., pp 847-855.

Beasley J.E. (1985). An Algorithm for the two-dimensional assortment problem. *Euro. J. of Ops. Res.* Vol 19, pp 253-261.

Becker W.E. and Harnett D.L. (1987). *Business and Economics Statistics with Computer Applications.* Addison-Wesley Publishing Co.

Bengtsson B. (1983). Packing rectangular pieces - A heuristic approach. *The Compt. J.* Vol 25, No 3, pp 353-357.

Bentley J.L., Haken D. and Hon R.W. (1980). Statistics on VLSI Designs. *Working Paper. Compt. Dept. Carnegie-Mellon Univ.* Pittsburgh PA. CMU-CS-80-111.

Bischoff E.E. and Marriott M.D. (1990). A comparative evaluation of heuristics for container loading. *Euro. J. of Ops. Res.* Vol 44(2), pp 267-276.

Böhme D. and Graham A. (1979). Practical Experiences with Semiautomatic and Automatic Partnesting Methods. *Compt. App. in the Automation of Shipyard Operation and Ship Design III.* Eds. Kuo C. MacCallum K. J. & Williams T.J. IFIP, N. Holland Pub. Co. pp 213-220.

Borland. (1991a). *Turbo C++ Second Edition - User's Guide.* Borland International, inc. Scotts Valley, CA 59067-0001, USA.

Borland. (1991b). *Turbo C++ Second Edition - Programmer's Guide.* Borland International, inc. Scotts Valley, CA 59067-0001, USA.

Burkard R.E. and Bonninger T. (1983). A heuristic for quadratic boolean program with applications to quadratic assignment problems. *Euro. J. of Ops. Res.* Vol 13, pp 374-386.

Catastini A., Cavagna C., Cugini U. and Moro P. (1976). A computer-aided design system for pattern grading and marker making in the garment industry. *Proceedings of CAD 76.* ICP 2496.09, pp 159-165.

Chambers M.L. and Dyson R.G. (1976). The Cutting Stock Problem in the Flat Glass Industry - Selection of Stock Sizes. *Operational Res. Quarterly.* Vol 27, No 4(ii), pp 949-957.

Christofides N. and Whitlock C. (1977). An Algorithm for Two Dimensional Cutting Problems. *Ops. Res.* Vol 25, No 1, Jan-Feb, pp 31-44.

Coffman E.G.Jr., Garey M.R., Johnson D.S. and Tarjan R.E. (1980). Performance bounds for level-oriented two-dimensional packing algorithms. *SIAM J. COMPUT.* Vol 9, No 4, Nov., pp 808-826.

Dagli C.H. and M. Tatoglu M.Y. (1987). An approach to two-dimensional cutting stock problems. *INT. J. PROD. RES.* Vol 25, No 2, pp 175-190.

De Cani P. (1978). A Note on the Two-dimensional Rectangular Cutting-stock Problem. *J. Operational Res. Soc.* Vol 29, No 7, pp 703-706.

Dori D. and Ben-Bassat M. (1984). Efficient Nesting of Congruent Convex Figures. *Comms. of the ACM, Image Processing and Computer Vision.* Vol 27, No 3, March, pp 228-235.

Dori D. and Ben-Bassat M. (1983). Circumscribing a Convex Polygon by a Polygon of Fewer Sides with Minimal Area Addition. *Computer Vision, Graphics and Image Processing.* Vol 24, pp 131-159.

Dowsland W.B. (1991). Three-dimensional packing - solution approaches and heuristic development. *INT. J. PROD. RES.* Vol 29, No 8, pp 1673-1685.

Dyckhoff H. (1981). A new linear programming approach to the cutting stock problem. *Ops. Res.* Vol 29, pp 1092-1103.

Flemming R. (1986). A computer system for small-batch sheet metal manufacture. *Computer-Aided Eng. J.* Vol 3, (2), pp 60-63.

Freeman H. and Shapira R. (1975). Determining the Minimum-Area Encasing Rectangle for an Arbitrary Closed Curve. *Comms. of the ACM.* Vol 18, No 7, July, pp 409-413.

Gardener M. (1979). Some packing problems which cannot be solved by sitting on the suitcase. *Scientific American.* October, pp 22-26.

Garey M.R. and Johnson D.S. (1979). Computers and Intractability - A guide to the theory of NP-Completeness. *W H Freeman and Co.* San Francisco.

Gehring H., Menschner K. and Meyer M. (1990). A computer-based heuristic for packing pooled shipment containers. *Euro. J. of Ops. Res.* Vol 44(2), pp 277-289.

George J.A. and Robinson D.F. (1980). A Heuristic for Packing Boxes into a Container. *Comput. & Ops. Res.* Vol 7, pp 147-156.

Gilmore P.C. and Gomory R.E. (1961). A Linear Programming approach to the Cutting Stock Problem. *Ops. Res.* Vol 9, pp 849-859.

Gilmore P.C. and Gomory R.E. (1963). A Linear Programming approach to the Cutting Stock Problem - Part 2. *Ops. Res.* Vol 11, pp 863-888.

Gilmore P.C. and Gomory R.E. (1965). Multi-stage Cutting Stock Problems in two or more dimensions. *Ops. Res.* Vol 13, pp 94-128.

Golan I. (1981). Performance Bounds for Orthogonal Oriented Two-dimensional Packing Algorithms. *SIAM J. Comput.* Vol 10, No 3, August, pp 571-582.

Golany B. and Rosenblatt M.J. (1989). A heuristic algorithm for the quadratic assignment formulation to the plant layout problem. *INT. J. PROD. RES.* Vol 27, pp 293-308.

Grunbaum B. and Shephard G.C. (1977). The eighty-one types of isohedral tilings in the plane. *Math. Proc. of the Cambridge Phil. Soc.* Vol 82, Part 2, pp 177-196.

Gupta T. and Ghosh B.K. (1988). A Survey of Expert Systems in Manufacturing and Process Planning. *Computers in Industry*, Vol 11, pp 195-204.

Haims J.M. and Freeman H. (1970). A multistage solution of the template-layout problem. *IEEE Trans. Syst. Sci. and Cybernetics.* Vol SSC-6, No 2, April, pp 145-151.

Hancock T.M. (1989). A Systems Approach to Sheet Metal Processing. *Computers in Industry.* Vol 13. pp 245-251.

Harnett D.L. and Murphy J.L. (1985). *Statistical Analysis for Business and Economics.* Third Edition. Addison-Wesley Publishing Company.

Herz J.C. (1972). Recursive Computational Procedure for Two-dimensional Stock Cutting. *IBM J. Res. & Dev.* Vol 31, Sept., pp 462-469.

Hinxman A.I. (1980). The trim-loss and assortment problems: A survey. *Euro. J. of Ops. Res.* Vol 5, pp 8-18.

Hodgson T.J. (1982). A Combined Approach to the Pallet Loading Problem. *IIE Trans.* Vol 14, No 3, Sept., pp 175-182.

Ikeda Y. (1979). The Pansy, an Advanced Interactive Parts Nesting System. *Compt. App. in the Auto. of Shipyard Operation and Ship Design III.* Eds. Kuo C. MacCallum K. J. & Williams T.J. IFIP, N. Holland Pub. Co. pp 313-321.

Illiev V.I., Popov G.S. and Ivanov M. (1989). A system for computer-aided preparation in the production of plane parts by stamping. *J. of Mech. Working Tech.* Vol 18, pp 283-292.

Ismail H.S. and Hon K.K.B. (1992). New approaches for the nesting of two-dimensional shapes for press tool design. *INT. J. PROD. RES.* Vol 30, No 4, pp 825-837.

Israni S. and Sanders J. (1982). Two-Dimensional Cutting Stock Problem Research: A Review and a New Rectangular Layout Algorithm. *J. Manf. Sys.* Vol 1, No 2, pp 169-182.

Israni S.S. and Sanders J.L. (1985). Performance testing of rectangular parts-nesting heuristics. *INT. J. PROD. RES.* Vol 23, No 3, pp 437-456.

Johnson D. S., Demers A., Ullman J. D. Garey M. R. and Graham R. L. (1974). Worst-case Performance Bounds for Simple One-dimensional Packing Algorithms. *SIAM J. Comput.* Vol 3, No 4, December, pp 299-325.

Kalpakjian S. (1989). *Manufacturing, Engineering and Technology.* Addison-Wesley Publishing Co., pp 452.

Kernighan B.W. and Ritchie D.M. (1978). *The C programming language.* Prentice Hall inc. Englewood Cliffs, New Jersey, USA.

Kershner R.B. (1968). On Paving the Plane. *American Math. Monthly.* Vol 75, pp 839-844.

Kleitman D.J. and Krieger M.M. (1970). Packing squares into rectangles; 1. *Annals of the New York Academy of Sciences.* Vol 175, Art 1, July, pp 253-262.

Kochan S.G. (1988). *Programming in C.* Hayden Books, Indianapolis, USA.

Koroupi F. and Loftus M. (1991). A Knowledge-Based Press Tool Assembly System. *INT. J. PROD. RES.* Vol 29, No 8, pp 1507-1519.

Kujanpää V. and Penttilä R. (1992). Integration of design, planning and manufacturing of sheet metal products. *Proc. of the Int. Conf. on Manufacturing Automation.* Held at the University of Hong Kong, August, pp 806-811.

Kusiak A. and Heragu S.S. (1987). The facility layout problem. *Euro. J. of Ops. Res.* Vol 29, pp 229-251.

Lascoe O.D. (1988). *Handbook of Fabrication Processes.* ASM International.

Lawler E.L. (1963). The quadratic assignment problem. *Management Science.* Vol 9, pp 586-599.

Linardakis S. and Mileham A.R. (1993). Manufacturing Feature Identification for Prismatic Components from CAD DXF Files. Proceedings of the 9th National Conference on Manufacturing Research held at the University of Bath, UK. September. *Published as Advances in Manf. Tech. VII.* pp 37-41.

Malmborg C.J. (1994). A heuristic model for simultaneous storage space allocation and block layout planning. *INT. J. PROD. RES.* Vol 32, No 3, pp 517-530.

Mansfield E. (1987). *Statistics for Business and Economics - Methods and Applications.* 3rd Edition. W.W. Norton & Co. New York, USA.

Mullish H. and Cooper H.L. *The Spirit of C - An Introduction to Modern Programming.* West Publishing Company, St. Paul, MN 55164-1003, USA, 1987.

Nee A.Y.C. - Sub. by Venkatesh V.C. (1984a). A Heuristic Algorithm for Optimum Layout of Metal Stamping Blanks. *Annals of the CIRP.* Vol 33, No 1, pp 317-320.

Nee A.Y.C. (1984b). Computer aided layout of metal stamping blanks. *Proc. Inst. Mech Engrs.* Vol 198b, No 10, pp 187-194.

Nee A.Y.C., Seow K.W. and Long S.L. - Sub. by Venkatesh V.C. (1986). Designing Algorithm for Nesting Irregular Shapes With and Without Boundary Constraints. *Annals of the CIRP.* Vol 35, No 1, pp 107-110.

Nee A.Y.C. (1989). PC-based computer aids in sheet-metal working. *Journal of Mechanical Working Technology,* Vol 19, pp 11-21.

Ngoi B.K.A., Tay M.L. and Chua E.S. (1994). Applying spatial representation techniques to the container packing problem. *INT. J. PROD. RES.* Vol 32, No 1, pp 111-123.

Nnaji B.O., Kang T., Yeh S. and Chen J. (1991). Feature reasoning for sheet metal components. *INT. J. PROD. RES.* Vol 29, No 9, pp 1867-1896.

Nolan B. (1994). *Data Analysis - An Introduction.* Polity Press, Cambridge, UK.

Page E. (1975). A Note on a Two-dimensional Dynamic Programming Problem. *Ops. Res. Qrtly.* Pergamon Press, Vol 26, No 2, i, pp 321-324.

Parry-Barwick S.J. (1995). Multi-Dimensional Set-Theoretic Geometric Modelling. *PhD Thesis.* University of Bath, Bath, UK.

Prasad Y.K.D.V. and Somasundaram S. (1991). CASNS - a heuristic algorithm for the nesting of irregular-shaped sheet-metal blanks. *Computer-Aided Eng. J.* April, pp 69-73.

Qiao L.H., Zhang C., Liu T.H., Wang H.P.B. and Fischer G.W. (1993). A PDES STEP-Based Product Data Preparation Procedure for Computer-Aided Process Planning. *Compts. in Industry.* Vol 21, No 1, pp 11-22.

Qu W. and Sanders J.L. (1987). A nesting algorithm for irregular parts and factors affecting trim losses. *INT. J. PROD. RES.* Vol 25, No 3, pp 381-397.

Qu W. and Sanders J.L. (1989). Sequence selection of stock sheets in two-dimensional layout problems. *INT. J. PROD. RES.* Vol 27, No 9, pp 1553-1571.

Raggenbass A. and Reissner J. (1989). Stamping-laser combination in sheet processing. *CIRP Annals.* Vol 38 (1), pp 291-294.

Raoot A.D. and Rakshit A. (1994). A 'fuzzy' heuristic for the quadratic assignment formulation to the facility layout problem. *INT. J. PROD. RES.* Vol 32, No 3, pp 563-581.

Roberts S.A. (1984). Application of Heuristic Techniques to the Cutting-Stock Problem for Worktops. *J. Operational Res. Soc.* Vol 35, No 5, pp 367-377.

Salisbury K. (1991). Set the controls for the heart of the sun. *Manufacturing Engineer.* Vol 70, No 7, pp 12-15.

Sanders J. L. and Bronsoiler A. (1983). Performance Testing of Irregular Parts Nesting Systems for Flame Cutting and other Industrial Applications. *Proc. 11th N. American Manf. Res. Conf.* (NAMRC-X1) held at Madison Wi. May, pp 434-440.

Schwarz A., Berry D.M., and Shaviv E. (1994). On the use of the automated building design system. *Computer-Aided Design.* Vol 26, No 19, October, pp 747-762.

Scott A.J. and Mileham A.R. (1992). A Code Based Process Planning System For Sheet Metal Components. *Proceedings of the Eight National Conference on Manufacturing Research.* Held at the University of Central England in Birmingham, UK. 8th-10th September. pp 63-74.

Scott A.J. and Mileham A.R. (1993a). Automated Nesting of Sheet Metal Components onto Stock Sheets. Proceedings of the 9th National Conference on Manufacturing Research held at the University of Bath, UK. September. *Published as Advances in Manf. Tech. VII.* pp 63-74.

Scott A.J. and Mileham A.R. (1993b). Automated Process Planning of Sheet Metal Components. *PhD Transfer Report.* School of Mech. Eng., University of Bath, UK, September.

Scott A.J. and Mileham A.R. (1994a). A New System to Nest Sheet Metal Parts Which Have Been Enclosed Within Rectangular Envelopes. *Proceedings of the Second International Conference on Sheet Metal* held at the University of Ulster, N. Ireland. 12th-13th April, pp 63-74.

Scott A.J. and Mileham A.R. (1994b). A New System to Nest Sheet Metal Parts in the 'low volume, high variety' environment. *Proceedings of the First International Conference on Manufacturing Processes, Systems and Operations Management in Less Industrialised Regions.* Held at the National University of Science and Technology. Bulawayo, Zimbabwe. 25th-27th April.

Scott A.J. and Mileham A.R. (1996). RECTNEST system code, test parts lists and results *Technical Report.* School of Mech. Eng., University of Bath, UK, June.

Shin W.S., Bullington S.F. and Causey D.W. (1990). A procedure for scheduling punch presses with restrictions on the sequence of punches. *INT. J. PROD. RES.* Vol 28, No 4, pp 723-734.

Sirinaovakul B. and Thajchayapong P. (1994). A knowledge base to assist a heuristic search approach to facility layout. *INT. J. PROD. RES.* Vol 32, No 1, pp 141-160.

Smith G.T. (1992). Go with the flow! An appraisal of CNC spinning & shear forming techniques. *Proc. of the Int. Conf. on Sheet Metal.* Held at Birmingham Polytechnic. pp 417-428.

Smith J.S., Cohen P.H., Davis J.W. and Irani S.A. (1992). Process plan generation for sheet metal parts using an integrated feature-based expert system approach. *INT. J. PROD. RES.* Vol 30, No 5, pp 1175-1190.

Sperling B. (1979). Nesting is more than a layout problem. *Compt. App. in the Auto. of Shipyard Operation and Ship Design III.* Eds. Kuo C. MacCallum K. J. & Williams T.J. IFIP, N. Holland Pub. Co. pp 287-294.

Stainton R.S. (1977). The Cutting Stock Problem for the Stockholder of Steel Reinforcement Bars. *Operational Res. Quarterly.* Vol 28, No 1(ii), pp 139-149.

Stamp R.J. and Earl C.F. (1992). Production of Sheet Metal Components by an Automatically Planned Robot Assisted Press Brake. *Proc. of the Int. Conf. on Sheet Metal.* Held at Birmingham Polytechnic. pp 211-219.

Svestka J.A., Dornfield D.A. and Gil R.A. (1981). On improving the productivity of numerically controlled punch presses. *INT. J. PROD. RES.* Vol 19, No 5, pp 471-480.

Svestka J.A. (1990). Imposing precedence constraints on NC punch press sequences. *INT. J. PROD. RES.* Vol 28, No 12, pp 2309-2319.

Tiow O.K., Wah P.W. and Guang W.C. (1992). Optimisation of materials processing using AutoCAD. *J. Mats. Proc. Tech.* Vol 29, pp 293-2999.

Tokuyama H. and Ueno N. (1985). The cutting stock problem for large sections in the iron and steel industries. *Euro. J. of Ops. Res.* Vol 22, pp 280-292.

Traister R.J. (1985). *Going from BASIC to C.* Prentice Hall inc. Englewood Cliffs, New Jersey, USA.

Turing A. (1936). On computable numbers with an application to the Entscheidungsproblem. *Proc. London Math Soc. Ser. 2.* Vol 42, pp 230-266 & Vol 43, pp 544-546.

Wang H. and Wysk R.A. (1987). Intelligent Reasoning for Process Planning. *Computers in Industry.* Vol 8. pp 293-309.

Wang P.Y. (1983). Two Algorithms for Constrained Two-Dimensional Cutting Stock Problems. *Ops. Res.* Vol 31, No 3, May-June, pp 573-586.

Wang W. and Bibb K. (1991). *Illustrated Turbo C++.* Wordware Publishing, Inc. Plano, Texas, USA.

Yuzu C., Lujun L., Wei W. and Jianwen S. (1987). An expert system for automatic allocation of 2D irregular shapes. *Proceedings of Expert Systems in Computer Aided Design Conference.* Elsevier Science Publishers B.V. (North Holland), IFIP, pp 407-422.

Zhang Y. and Mileham A.R. (1990). A CAD Interpreter for Interfacing CAD and CAPP of 2-$^1/_2$D Rotational *Parts. Proceedings of the Sixth International Conference on CAD/CAM, Robotics etc.* Held at Southbank Polytechnic. pp 333-336.

Abbreviations.

1-D	-	One Dimensional.
2-D	-	Two Dimensional.
$2\text{-}^1/_2D$	_	Two and a half Dimensional.
3-D	-	Three Dimensional.
BL	-	Bottom Left (Nesting Algorithm).
CAD	-	Computer Aided Design.
CAPP	-	Computer Aided Process Planning.
CPU	-	Central Processing Unit.
DXF	-	Drawing Interchange File.
IGES	-	Initial Graphics Exchange Specification.
LP	-	Linear Program.
PC	-	Personal Computer.
PPS	-	Production Planning System.
NC	-	Numerical Control.

Appendix A

Pseudocode Functions

This appendix contains the details of the pseudocode functions referred to in section 4.3. Each function title in this appendix is followed by an alpha-numeric code which corresponds to the codes of the function in Section 4.3. The function in this appendix are listed in alphabetical order on the basis of the code, not in the order in which they appear in Section 4.3. Pointers, variable names and examples of code are shown in bold typeface.

SEARCH FOR A PERFECT FIT PART (**A1**)

This function aims to identify a part which will fit the current space to fill almost perfectly. A pointer **best_part** is initially set to **no_part_found**. The program loops through the list until a part is found which, with its associated bridge gap, will fit within the space. The address of this part replaces **no_part_found** in **best_part**. For the rest of the search, if a part is found which will fit within the space and its area is greater than that of the current **best_part**, the pointer will be reset to this part's address. When every remaining part in the list to nest has been compared to the space to be nested the **best_part** pointer is examined. If a part is held in **best_part** and its area is greater than 90% of the area of space, the part is returned by the function. Otherwise **best_part** is re-set to **no_part_found** and returned to the calling function. The latter indicates that a perfect fit part cannot be found and another method of placement must be used. If part rotation is permitted this process is carried out considering the part in both of its primary orientations.

SEARCH FOR A DIMENSION FIT PART (A2)

This function aims to identify a part which will closely fit one of the dimensions of the current space to nest. A pointer **best_part** is initially set to **no_part_found.** The program loops through the list until a part is found which, with its associated bridge gap, will fit within the space. The address of this part replaces **no_part_found** in **best_part.** For the rest of the search, if a part is found which will fit within the space and its X dimension is greater than that of the current **best_part,** the pointer will be reset with its address. When every remaining part in the list to nest has been compared to the space, the **best_part** found is made the **total_best_part** and the **best_part** pointer is re-set to **no_part_found.** This search procedure then is repeated for the Y dimension. If the Y dimension of this **best_part** found is not closer to the space Y dimension than the **total_best_part** X dimension is to the space X dimension the settings remain the same. Otherwise the **total_best_part** is given the address of this **best_part.** If the chosen part dimension is within 90% of the space dimension the **total_best_part** is returned to the calling function. Otherwise **total_best_part** is re-set to **no_part_found** and returned; this indicates that a dimension fit part cannot be found, which will require the bottom left method of placement to be tried, see function (A4). If rotation is permitted this is carried out for both part orientations.

ADJUST PRIMARY AREA FOR DIMENSION FIT (A3)

If the difference between the part's X dimension and the space's X dimension is less than the difference between the part's Y dimension and the space's Y dimension, the Y dimension of the space is reduced by the part's Y dimension and its associated bridge gap. The space's Y co-ordinate must be increased by this value. Otherwise the X dimension of the space is reduced by the part's X dimension, and its associated bridge gap, and the space's X co-ordinate is increased accordingly. This adjusted space remains the primary area to nest in the calling function.

SEARCH FOR A BOTTOM LEFT FIT PART (A4)

This function aims to identify the best part to place in the bottom left of the current space (primary area to nest), following which this space will be split into two, as shown in Figure A.1. The part should be chosen to maximise the ease of subsequent part nesting. If the space's X dimension is smaller than its Y dimension the space will be split by a horizontal line from the placed part's top right corner. By using the part with the largest Y dimension in the list which will fit within the space, any of the remaining parts can be placed in this area. This promotes the placing of the largest parts in the list and maximise the choice of parts to place. If the space's Y dimension is the smaller, the part with the largest X dimension will be used. If part rotation is permitted the part with the largest dimension (which the space can accommodate) is selected and placed in the required orientation.

To find the part to place a pointer **best_part** is initially set to **no_part_found.** The program searches through the list to find the first part which, with its associated bridge gap, will fit within the space. The address of this part replaces **no_part_found** in **best_part** and the rest of the list of parts is searched. If any part is found which will fit within the space and has a larger chosen dimension it is made the best part. If no suitable part is found **best_part** will be returned to the calling function set to **no_part_found.** As this is the final method of placement in the Area Fit system, this will cause the space to be discarded as waste.

BOTTOM LEFT SPLIT PRIMARY AREA (A5)

In this situation the part is being placed in the bottom left corner of a relatively large space. To maintain the rectangular representation, the existing space must be split into two rectangular spaces by drawing a line from the top right corner of the placed part. Figure A.1 shows the two possible area splits. If the line is drawn vertically the resulting spaces have the following areas:

space_1_area = part->x_dim * (old_space->y_dim - part->y_dim);

space_2_area = (old_space->x_dim - part->x_dim) * old_space->y_dim;

If the line is drawn horizontally the resulting spaces have the following areas:

space_1_area = (old_space->x_dim - part->x_dim) * part->y_dim;

space_2_area = old_space->x_dim * (old_space->y_dim - part->y_dim);

If the space's X dimension is smaller than its Y dimension the space will be split by a horizontal line from the placed part's top right corner. Otherwise a vertical division line will be used. The old area is replaced with these two new areas which are ordered with the smaller as the primary area to nest. This maintains the largest possible secondary area and prioritises positioning parts into the smaller spaces on the sheet.

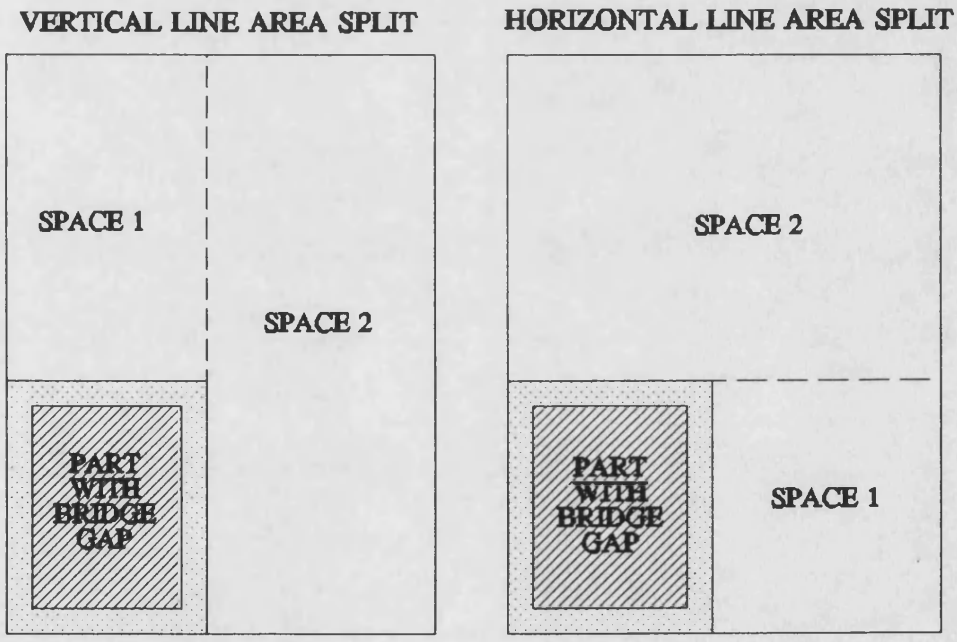


Figure A.1.
Possible Bottom Left placement area splits.

FIND AN END AREA FOR THE NEXT PART (F1)

The end areas are held in a linked list and are separate from the normal areas to nest. The pointer to the start of the end area list is initially set to a flag, showing that the list is empty, which the first end area created will replace. Placing parts in end areas is preferable to placing parts in the conventional areas as any end areas remaining when the layout is complete will automatically be considered as waste. Parts are placed within the first end area found which will accommodate them. A pointer **best_end_area** is pre-set to a flag value **no_end_area**. The program searches through the end area list and the address of the first end area found which will accommodate the part will replace **no_end area** in **best_end_area**. If a suitable end area has been found or the complete end area list has been searched the **best_end_area** is returned to the calling function. If its value is still set to **no_end_area** the calling function will be flagged to try to place the part in a normal area on the sheet.

ADJUST THE END AREA AFTER PART PLACEMENT (F2)

The First Fit algorithm, which uses this function, places parts by strip placement ie. the remaining strip area is reduced only in its Y dimension. Following this convention is necessary to allow subsequent application of the Strip Squeeze algorithm and also maintains the homogeneity of the system. Thus any free space at the side of the placed part is ignored and the end area is reduced in the Y dimension by the placed part's Y dimension and associated bridge gap. End areas which are too small to accommodate any part are not removed from the list as the additional complexity of doing this does not justify the small saving in the time to search the list of end areas.

MAKE PRIMARY AREA AN END AREA (F3)

This function is called when the top part in the list to nest will not fit within the remaining area at the end of the strip being formed. It is desirable to save this area, as a later part in the list may fit within it. This is done by copying the dimensions and

co-ordinates of the remaining primary area into an end area structure which is placed in a separate linked list. Each new end area added to the list is placed at the top. When the Strip Squeeze algorithm is applied to the last two strips placed, their associated end areas must also be adjusted. The end areas to adjust during the strip squeezing process are easily identified as they hold the first two positions in the linked list of end area structures.

START A NEW SHEET (G1)

When a new sheet on which parts are to be nested is started, the dimensions of the primary area to nest must be set to the sheet dimensions, minus the bridge gap. If this is the first sheet to be nested the first structure in the list of nested parts must be created and set as a sheet identifier. This is done by setting the **id_num**, **x_pos** and **y_pos** values of this structure to zero and the **x_dim** and **y_dim** values are given the sheet dimensions. If a sheet has already been nested the old primary area values must be overwritten with the bridge gap adjusted new sheet dimensions. Also, a new nested part structure will have to be added to the end of the output list with its values set as a new sheet identifier.

SET FOR SQUEEZE (G2)

A dedicated pointer **pre_strip_ptr** is given the address of the last part placed in the nested parts list. The next part placed in the list will be the first part in the first strip to squeeze. The Strip Squeeze algorithm uses **pre_strip_ptr** to identify the two strips to squeeze. When the Strip Squeeze algorithm has been used **pre_strip_ptr** is set to the last part in the second strip. This part will precede the first part of the first strip of the next pair of strips to squeeze. If a new sheet is started **pre_strip_ptr** is set to the new sheet identifier in the nested parts list.

PLACE PART (G3)

When a part is placed it must be removed from the list of parts to be nested and placed within the output list of parts. A new **nested_part** structure is created and the **id_num, x_dim** and **y_dim** are copied directly from the structure of the part to be nested. The co-ordinates in the nested part structure, **x_pos** and **y_pos**, are set to the co-ordinate values of the area with the associated bridge gap added. This **nested_part** structure is placed at the end of the output list. Finally the **part_to_nest** structure is removed from the input list and its allocated memory is returned to 'free' status.

MAKE SECONDARY AREA PRIMARY (G4)

The spaces to be nested on the sheet are held in a linked list in order of increasing area, to preserve the maximum free sheet area. If the first (primary) area in the list is too small to accommodate any remaining part to nest, it must be discarded as waste area and the next area in the list considered. This is done by removing the first area and freeing its memory allocation. The list pointers are altered to maintain a closed list and the list's external pointer is set to the address of the new primary area.

REDUCE THE PRIMARY AREA BY STRIP PLACEMENT (G6)

When a part is placed in a strip the remaining strip area is reduced only in its Y dimension, the equations to do this are shown below. Any free area at the side of the part is ignored and accepted as waste. If the squeeze algorithm is to be used much of this waste area may be recovered.

area->y_dim = area->y_dim - part->x_dim + bridge_gap;

area->y_pos = area->y_pos + part->x_pos + bridge_gap;

SEPARATE THE STRIPS FROM THE LIST OF PARTS (Q1)

The **pre_strip_ptr**, see function (G2), indicates the part in the output list which precedes the first strip to be squeezed. A pointer **start_strip_1** is set to the address of the part after **pre_strip_ptr**. All the parts within a strip will share the same X co-ordinate, so the list is searched from **start_strip_1** and the address of the first part encountered with a different X co-ordinate is set as **start_strip_2**. The list search is continued to find the last part in the second strip, the pointer of which will be set to the first part in the output list. The pointer of this last part in the second strip is set to **start_strip_2** and the pointer of the last part in the first strip is set to **start_strip_1**; this closes the strip lists. The pointer of the part identified by **pre_strip_ptr** is set to the first part of the output list, closing the output list. Thus the strips to squeeze have been separated into two linked lists.

SORT STRIP INTO X DIMENSION ORDER (Q2)

The list pointer, **start_strip_1** or **start_strip_2**, are re-set to the part with the largest X dimension. The list is the repeatedly searched for the next largest part, which is re-positioned below the parts which have already been X dimension sorted.

PLACE PARTS FROM THE BOTTOM LEFT (Q3)

The parts are now in X dimension sequence, but their co-ordinates still reflect their old positions. The parts must therefore be re-positioned into the X dimension sequence. The first part in the list has its Y co-ordinate set to the strip area Y co-ordinate, **strip_y**, which will initially be the appropriate bridge gap. The Y dimension of the first part, with the appropriate bridge gap, is then added to **strip_y** and the Y co-ordinate of the next strip part is the set to this value. This is repeated for all the strip parts. This is effectively strip placement, see function (G6), however as the parts are in a group known to fit into the strip area, it is not necessary to check the positions of the parts relative to the sheet's bounds.

PLACE PARTS FROM THE TOP RIGHT (Q4)

The parts are now in X dimension sequence, but their co-ordinates still reflect their old positions. The parts must therefore be re-positioned into the X dimension sequence, however unlike function (Q3) this is to be done from the top right of the sheet area. The first part in the list has the largest X dimension and therefore is the strip width; from this the value **strip_x_dim** is set. The current X co-ordinates of the parts are all the same, **strip_x_pos** is set with this value. The X co-ordinates of all the parts in the second strip are then re-set by the following equation.

part->x_pos = strip_x_pos + (strip_x_dim - part->x_dim);

The value **strip_y** is initially set to the sheet's Y dimension, with no account for the bridge gap. Every part in the list is then re-positioned in the Y axis. Each time a part is placed its Y co-ordinate is re-set according to the equation below and **strip_y** is reduced by the part's Y dimension and the appropriate bridge gap.

part->y_pos = strip_y - part->y_dim - bridge_gap;

As with function (Q3) the parts are in a group known to fit into the strip area so it is not necessary to check the positions of the parts relative to the sheet's bounds.

FIND THE GAPS AT TOP RIGHT PART CORNERS (Q5)

This function establishes the gap between the top right corner of a part in the first strip and the part in the second strip which is opposite this corner, see Figure A.2. The function is repeated for all the parts in the first strip and the smallest gap found is retained. Initially the part in the second strip which is opposite the corner of the part in the first strip must be searched for. The conditions for a part in the second strip, **part_2,** being opposite the part in the first strip, **part_1**, are as follows (**>=** signifies greater than or equal to and **bg** signifies the bridge gap).

part_1->x_pos+part_1->x_dim+(bg/2) >= part_2->x_pos-(bg/2); AND

part_1->x_pos+part_1->x_dim+(bg/2) < part_2->x_pos+part_2->x_dim+(bg/2);

The width of each strip (**strip_1_x_dim** and **strip_2_x_dim**) is equal to the largest X dimension which it contains. The gap between the strips at this point is as follows.

gap = strip_1_x_dim + strip_2_x_dim - part_1->x_dim -part_2->x_dim -bg;

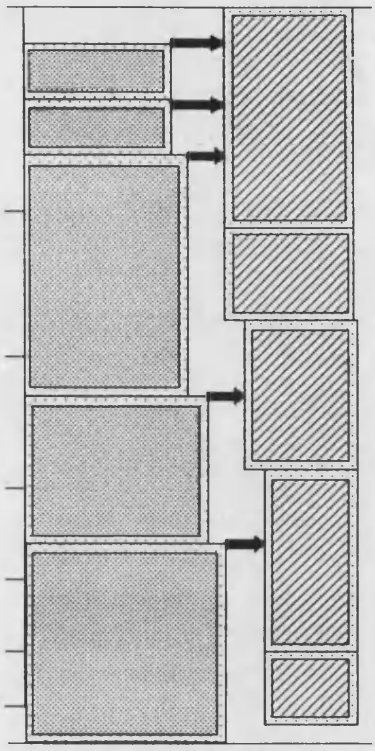


Figure A.2.
Top right gaps.

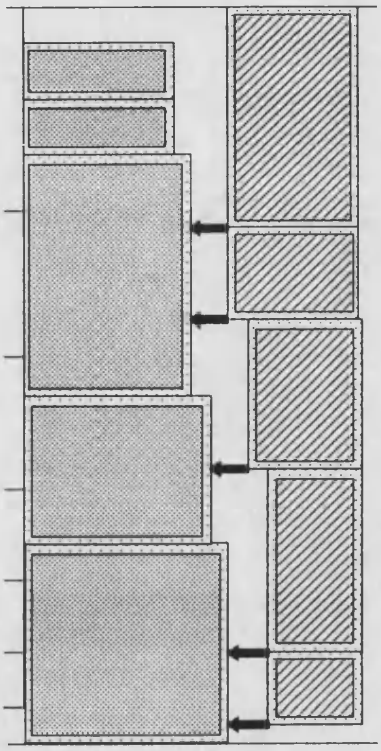


Figure A.3.
Bottom left gaps.

FIND THE GAPS AT BOTTOM LEFT PART CORNERS (Q6)

This function establishes the gap between the bottom left corner of a part in the second strip and the part in the first strip which is opposite this corner, see Figure A.3. The function is repeated for all the parts in the second strip. Any gap found which is smaller than the current smallest gap is retained. Initially the part in the first strip which is opposite the corner of the part in the second strip must be searched for. The conditions for a part in the first strip, **part_1**, being opposite the part in the second strip, **part_2**, are as follows (**>=** signifies greater than or equal to and **bg** signifies the bridge gap).

part_2->x_pos-(bg/2) >= part_2->x_pos-(bg/2); AND

part_2->x_pos-(bg/2) < part_1->x_pos+part_1->x_dim+(bg/2);

The width of the strips, **strip_1_x_dim** and **strip_2_x_dim**, are the X dimensions of the largest part which each contains. Thus the gap between the parts, by which they could be squeezed, is as follows.

gap = strip_1_x_dim + strip_2_x_dim - part_1->x_dim -part_2->x_dim -bg;

LEFT TRANSLATE STRIP 2 PARTS BY THE GAP (Q7)

To create the layout improvement the parts in the second strip must be translated to the left by the squeeze gap. Each part in the second strip is taken in turn and the squeeze gap is subtracted from its X co-ordinate.

CHANGE THE REMAINING AREA (Q8)

Due to the parts in the second strip being translated to the left, the remaining area on the sheet is increased. This is accounted for by adding the squeeze gap to the X dimension of the primary area and subtracting the squeeze gap from the X co-ordinate of the primary area.

CHANGE THE APPROPRIATE END AREAS (Q9)

This function will only be used if the Strip Squeeze algorithm is being applied to the First Fit algorithm. Both end areas must have their X dimensions reduced by the squeeze gap used for the parts. In addition, the co-ordinates of the second strip's end area must be altered. The Y co-ordinate will be set to the value of the bridge gap and the X co-ordinate must be reduced by the squeeze gap.

RETURN THE STRIPS TO THE LIST OF NESTED PARTS (Q10)

The individual strips to squeeze were separated from the main list of nested parts by function (Q1). The **pre_strip_ptr** indicates the last part in the output list, after which the squeezed strips must be added. The pointer of the **pre_strip_ptr** part must be re-set to the address of **start_strip_1**. The last part in the first strip can easily be found as its pointer will be set to the address of **start_strip_1** to complete the first strip's closed list. This pointer must be re-set to the address of **start_strip_2** to include the second strip in the output list. The last part in the second strip can also be easily found as its pointer will be set to the address of **start_strip_2** to complete the second strip's closed list. The pointer of this last part in the second strip must be set to the first part in the list of nested parts, creating a closed output list.

Now that the strips have been replaced in the output list, the **pre_strip_ptr** must be set to the last part of the second strip, as in function (G2). This allows the next pair of strips placed to be identified and passed to the Strip Squeeze algorithm. This completes the operation of the Strip Squeeze algorithm allowing the original part placing algorithm to recommence its operation.

RELAX THE STRIP ACCEPTANCE TOLERANCES (S1)

The initial strip width acceptance tolerance is 90% and the strip length acceptance tolerance is 90%. This function alternately relaxes the width tolerance and length tolerance, with the width tolerance being relaxed first. At each relaxation of the strip width acceptance tolerance the accepted band is doubled ie. the second width tolerance used would be 80% and the third 60%. The strip length acceptance tolerance is decremented each time by 5% making the second strip length acceptance tolerance 85% and the third 80%. If the values should be relaxed by these equations to less than zero their values will be re-set to zero. Acceptance tolerances of zero would mean any remaining parts could be combined into a strip of any length. This is essential to ensure that the entire list of parts can be placed.

These relaxation values were established by trial and error. The strip width acceptance tolerance is relaxed faster than the strip length acceptance tolerance since, if the Strip Squeeze algorithm is to be applied, some of the waste incurred can be recovered. It must be noted that this algorithm always tries to maximise the strip length from the available parts before it is tested for conformity to the strip length tolerance. Thus the strip length achieved usually exceeds the strip length tolerance imposed by a considerable amount.

SET NEW STRIP WIDTH LIMITS (S2)

Initially the strip width tolerance is assumed to be 90% in this section and parts are assumed to be of fixed orientation. For the first loop the strip width upper limit is set as the largest X dimension in the parts list and the lower width limit is $^{9}/_{10}$ths of this value. On each subsequent loop, the strip width upper limit is set as the largest X dimension found in the list which is smaller than the previous strip width upper limit. The program will loop until either every part has been placed or every X dimension in the list has been tried. This latter condition is achieved when the strip width upper limit is equal to the smallest X dimension in the list. If part rotation is permitted the Y dimensions of parts will also be searched for the next largest dimension to base the strip width upper limit on. The strip length tolerance is held at 90%, therefore any strips of parts with a length of greater than $^{9}/_{10}$ths of the sheet side length are accepted. These acceptance limits have been established by trial and error and are discussed in more detail in the tolerance relaxing function (S1).

SET END OF SHEET STRIP WIDTH LIMITS (S3)

At the end of the current sheet the X dimension of the remaining area may be less than the strip width upper limit, which is based on the parts to be nested, see function (S2). Rather than waste this area, the strip width upper limit is temporarily set to the width of this area, allowing a suitable strip to be formed. The conventional strip width upper limit (derived from the dimensions of the parts) will be retained, as it will be

appropriate for the new sheet to be used. As usual, the lower strip width limit will be set at $^{9}/_{10}$ths of the upper limit.

COLLECT ALL THE PARTS WITHIN THE WIDTH LIMITS (S4)

This function aims to create a sub-group containing all the parts with an X dimension within the current strip width limits. If parts do not have a fixed orientation their Y dimensions will also be considered. Rather than separating all the relevant parts from the main list of parts to nest, a new **strip_part** structure is used which contains a pointer to the actual part's structure. This **strip_part** structure also contains the dimension of the part which will lie along the Y axis of the prospective strip. This allows the parts' Y dimensions to be considered in numerous combinations for strip formation without the real part structures being altered. The function searches through the input list considering each part. If the current part has an X dimension within the strip width bounds a new **strip_part** structure is created and the part's address and Y dimension are entered. If part rotation is permitted and the part's Y dimension is within the strip width bounds, the real X dimension value is entered into the **strip_part** structure's Y dimension. These structures are placed in a linked list.

The Y dimension values of all the **strip_part** structures formed are summed, with their associated bridge gap, and the total compared to the lower strip length limit. If the summed value is greater than the lower length limit for the strip, ensuring that there are enough parts to potentially form a suitable length strip, the **strip_part** list is returned to the calling function. If the Y dimension total is less this length limit the linked list of **strip_part** structures is deleted. This has no effect on the linked list of structures containing parts to nest. A flag is also returned to the calling function to indicate that a suitable strip group cannot be formed.

TRY TO FORM A STRIP GROUP (S5)

This function inherits a **strip_part** list from function (S4) which is known to contain enough parts to span the strip length. Occasionally all the parts in the **strip_part** list, if combined, will create a strip within the strip length upper and lower limits. However, generally the **strip_part** list will contain too many parts for all to be placed in the strip and a combination of the member parts must be identified which will form a suitable strip. An exhaustive search of all possible combinations could be used, as this is effectively a one dimensional linear programming problem. However, this approach may prove time consuming with a large **strip_parts** list and it would not prioritise the positioning of large parts. In fact, as more small parts are required to make a strip of a set length and a greater number of parts gives a greater number of possible permutations, the inclusion of the large parts would effectively be discouraged. Thus a truncated search of the possible combinations is used which encourages the use of the large parts in the **strip_part** list.

The first step is to sort the **strip_part** list into decreasing Y dimension order. A **base_strip** is then formed by strip placing the parts in Y dimension sequence. Each time a new part is added to the **base_strip**, while the Y dimension sum does not exceed the strip length upper limit, the strip part combination so far is made the **best_strip**. When the part which causes the sum length to exceed the upper limit is placed, the **base_strip** is complete. The **strip_part** list is now split into **base_strip** parts and remaining parts. All the **base_strip** parts will be larger than any of the remaining parts. This is shown in Figure A.4.

The first **base_strip** part becomes the substitute part and every remaining part is tried in its place. If any combination is found with a sum length which is less than the strip length upper limit and which exceeds the length of the **best_strip**, the set of parts within **best_strip** are replaced with this new combination. This process is repeated until every other part in the **base_strip** has been the substitute part. The number of iterations which this process passes through is equal to the product of the number of parts in the base strip and the number of remaining parts. For example, a base strip

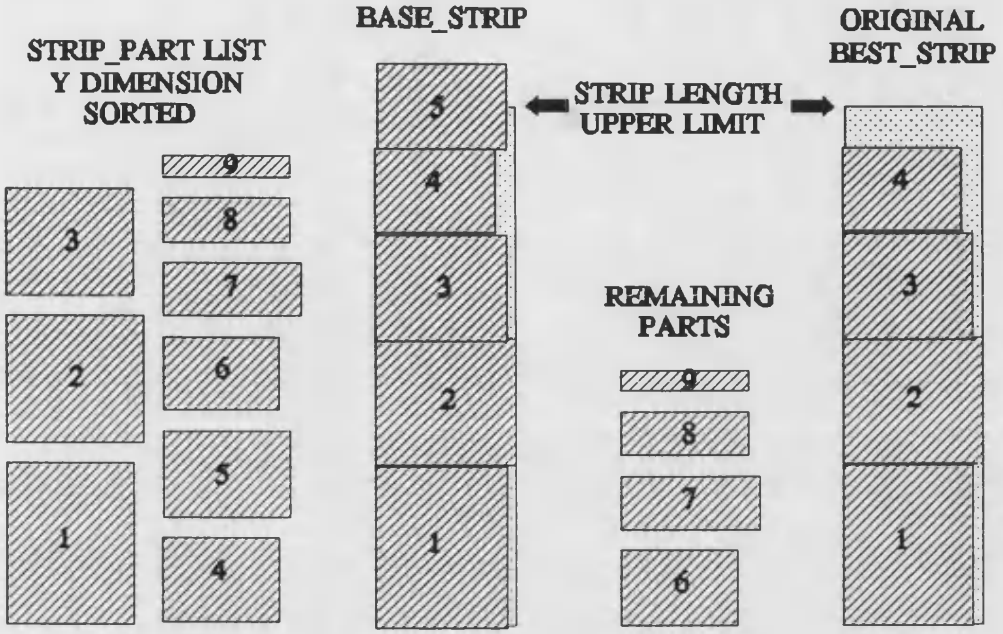


Figure A.4.
The groups of parts used by the Strip Fit algorithm.

of five parts, with four remaining parts to substitute in, would only require 20 iterations. If the sum length of the **best_strip** exceeds the strip length minimum limit, the remaining parts (not members of the **best_strip**) are deleted from the **strip_part** list and the **best_strip** list can then also be deleted. The remaining **strip_part** list is then returned to the calling function for part placement. If the sum length of the **best_strip** does not exceed the strip length minimum limit, the entire **strip_part** list is deleted and a flag is returned to the calling function to indicate that no suitable combination of parts could be found.

COLLECT ALL PARTS NARROWER THAN THE AREA (S6)

This function aims to create a sub-group containing all the parts with an X dimension less than the width of the remaining area at the end of the sheet. This function is only used when creating a strip at the end of a sheet when the remaining area is too narrow to accommodate a conventionally formed strip. If parts do not have a fixed orientation their Y dimensions will also be considered. As in function (S4), a new **strip_part**

structure is used which contains a pointer to the real part structure. This **strip_part** structure also contains the dimension of the part which will lie along the Y axis of the prospective strip. This allows the parts' Y dimensions to be manipulated without the real part structures being altered. For each part found which will fit in the end area a **strip_part** structure is created and the part's address and Y dimension are entered. If part rotation is required and permitted and the part's X dimension is entered into the **strip_part** structure's Y dimension. These structures are placed in a linked list which is returned to the calling function regardless of the potential strip length of the parts.

FORM AN END STRIP GROUP (S7)

This is a contingency function which creates a strip in a sheet end area when the conventional strip forming method, function (S5), has failed. The **strip_part** list is generated by function (S6), which includes any part which will fit within the area. The first step is to sort the **strip_part** list into decreasing X dimension order. Each **strip_part** is then placed in this sequence into the strip area. When a **strip_part** is too large for the remaining area it is deleted from the **strip_part** list and the next part considered. This function is only called if parts have been found which will fit into the end strip area and therefore, as there are no acceptance bounds with this strip, it will always form a strip. The **strip_part** list is then returned to the calling function for placement.

RE-SET NORMAL STRIP WIDTH LIMITS (S8)

To deal with the nesting of the area at the end of the sheet the strip width upper limit was re-set to the X dimension of the remaining area, function (S3). This function (S8) is positioned after the functions to place parts in the end of sheet area and re-sets the strip width upper limit to its previous value, which is required for the nesting of parts on the new sheet.

REMOVE STRIP GROUP PARTS FROM THE INPUT LIST (S9)

If functions (S5) or (S7) have returned a **strip_part** list, the structures of the parts to nest can be removed from the main list of parts to nest. These structures will not be lost as their addresses are held in the pointers of the **strip_part** structures. The **strip_part** list is stepped through and the associated part to nest structure is removed from the input list and the pointers of its neighbouring structures re-set.

ROTATE THE NECESSARY PARTS (S10)

If the parts are not held to a fixed orientation, many of the parts to nest held in the **strip_part** list may require rotation prior to placement on the sheet. This is the case if the Y dimension held in the **strip_part** structure is equal to the X dimension of the actual part, held in a **part_to_nest** structure. This check is made for all the parts in the **strip_part** linked list. If rotation is required the X and Y dimensions in the **part_to_nest** structure are swapped.

Appendix B

Statistical Tests

Pearson's Measure of Skewness (Sk_p) is calculated using the following formula:

$$Sk_p = 3(\text{Mean} - \text{Median}) / \text{Standard Deviation}$$

This value cannot have a magnitude of greater than 3. A curve is considered to be moderately skewed if its Pearson's skewness value has a magnitude of less than 1.

The Mann-Whitney Test is used to determine whether two samples are drawn from the same population. The two samples, containing n_1 and n_2 members respectively, are combined and each member is ranked according to its position within this combined group. If two or more members have the same value their ranks are averaged. The rank score r_1 is calculated for either sample set by summing the ranks of its members. The Mann-Whitney statistic **U** is then calculated using the following formulas:

$$T_1 = n_1n_2 + (n_1(n_1 + 1))/2 - r_1$$

$$T_2 = n_1n_2 - T_1$$

$$\text{If } T_1 <= T_2 \text{ } U = T_1 \text{ ELSE } U = T_2$$

Tables for the Mann-Whitney test give critical **U** values for different levels of significance (α). A significance value (α) of 0.05 means we can be 95% confident that the samples are from different populations if the **U** value is less than the critical value in the table.

Appendix C

Published Papers

1. Scott, A.J., Mileham, A.R. "A Code Based Process Planning System for Sheet Metal Components", Proc.8th Nat.Conf.on Manf.Research, pp 16-20, Univ of Central England, Sept 1992.

2. Scott A.J., Mileham A.R.," Automated nesting of sheet metal components onto stock sheets", Advances in Man.Tech VII, ed. by A.Bramley & A.R.Mileham, Bath University Press, Proc 9th NCMR Conf, September 1993, pp.242-246. ISBN No 1 85790 007.

3. Scott A.J. and Mileham A.R. " A New System to Nest Sheet Metal Parts Which Have Been Enclosed Within Rectangular Envelopes", Proceedings of the Second International Conference on Sheet Metal held at the University of Ulster, N. Ireland. pp 63-74. 12th-13th April 1994.

4. Scott A.J. and Mileham A.R., "A New System to Nest Sheet Metal Partss in the 'low volume, high variety' environment", Proceedings of the First International Conference on Manufacturing Processes, Systems and Operations Management in Less Industrialised Regions, National University of Science and Technology. Bulawayo, Zimbabwe. 25th-27th April 1994. Paper 2.2

A Code Based Process Planning System For Sheet Metal Components

A.J. Scott and A.R. Mileham

School Of Mechanical Engineering, University Of Bath.

Abstract.

This paper describes the preliminary elements of a process planning system for sheet metal components. The system is being developed around a component coding structure, with a component's code reflecting the geometric features which are critical for process selection. Thus this coding system differs in use and purpose from a design retrieval code, which groups together components of similar size and shape. As the new code identifies the process critical features in a component it aids process planning and should be particularly useful for group technology applications. The system aims to cover every possible sheet metal component. Although this will be too great a range for the practical needs of most companies, it is considered easier to remove unwanted sections than add in additional sections to a less comprehensive system.

Introduction.

Many existing sheet metal classification systems, such as the Opitz system 'Opitz, 1970', concentrate on geometric shape for design retrieval, rather than process relevant features. There are also process planning systems for sheet metal which specialize on a limited area, such as bend sequencing 'Hoffmann et al, 1992'. A coding system developed at the University of Virginia for the Lofton Corporation 'Kimpel and Richards, 1991' is comprehensive for cutting and bending, however Lofton's product range does not require formed features to be considered. Thus there is a need for a comprehensive process planning system and a classification system which covers the entire range of sheet metal components. Such a classification and coding system has been developed by the author and forms the basis of this paper. As the classification is process based it should be particularly useful for group technology applications.

The code structure is a simple and convenient vehicle for planning system development allowing easy manual use. Process planning rules can be established and manipulated without the complexity of computerised feature interpretation or processing. Process selection for a

component can be divided into two stages. Initially the processes which are capable of producing a feature must be identified, then the most economic of these must be selected. In its research form the component's code is limited to identifying the processes which are technically capable of producing the required features i.e. the first stage of process selection. Selection of the optimum process route would require far more information, such as process availability, process costs, scale of production, material type, sheet thickness etc.

Before grouping sheet metal components, it is necessary to establish their range and diversity. One method would be to examine all the sheet metal components currently produced. However this would be time consuming, some component types may be missed in the search and new components are constantly being designed. A simpler and quicker method is to study all the forms and features which can be created using the processes currently available. If the classification system is then designed to accept components containing any viable combination of these features it should comprehensively cover the entire range of sheet metal components.

Structure of the coding system.

There are two main types of code structures, Hierarchical and Attribute 'Bedworth et al, 1991'. In an attribute code each part attribute is assigned to a fixed position in a code. The meaning of each character in the code is independent of any other character value. In a hierarchical code the meaning of each character is dependant on the meaning of the previous character in the code. As no natural hierarchy exists within the range of sheet metal components and features an attribute code format has been used.

The processes for sheet metal fall into three clear categories, cutting, forming and bending. The features in each category are almost completely independent of the other categories. Exceptions include the interference of one category's feature during the manufacture of another category's feature and the order in which features are manufactured. Account can be made for these factors at a later stage in the system's development. Thus for process planning coding the categories can be considered autonomous and treated independently. In each code category the aim is to identify the features which correspond to the process required to manufacture the particular features. At this stage many factors are ignored, such as the number, orientation and position of features. The only factors considered are the component's feature types as this governs which processes can be used.

Cutting is the least complex catagory as it is constricted to two dimensions. In general only one process type will be used for all the cutting required on a component, allowing decisions to be based on the most complex features found in the component. For example, a low volume component with complex contours could be completely cut using CNC laser or plasma, regardless of straight edges which are within the process capability of a cheaper shearing process. This would be economic due to the hidden costs of complicating the process route. To take advantage of this, the cutting section considers three levels of complexity for internal features and three levels of complexity for perimeter features (see Tables 1 & 2).

Bending and forming processes are feature dedicated requiring the specific feature types found in a component to be identified. Thus the forming and bending sections of the system consists

of critical features which are either present in the subject component or not. This format is ideally represented by a binary code, as each digit can represent a feature type, with a '1' entered if it is present or a '0' if it is not (see Tables 3 & 4). Although the perimeter cutting and internal cutting elements of the code do not naturally conform to a binary code format, they will each have a two digit binary response to maintain the system's homogeneity. If the system is computerised the binary code generated can be directly stored in the memory.

The code's binary format is demonstrated in figure 1 by classifying the component in figure 4. The '1 0' response in the perimeter cutting section reveals that it consists of straight edges with internal angles, and the '0 0' response in the internal cutting section reveals that there are no internal features. The first digit of the folding section is a '1' indicating that folds are present and the third digit is also a '1', thus the only complicating factor is axial interference. As all the digits in the last section register '0' the component has no formed features.

Conclusion.

Product features critical for process selection in sheet metal planning have been identified and categorised under cutting, forming or bending. A unique classification and coding system has been devised which is based on these critical features. This allows the code to remain compact with no overlaps or omissions. The code used is a three stage, 15 digit binary code that encompasses all sheet metal components and is considered to significantly assist in the identification of the possible processes required to produce sheet metal components.

Further work.

Currently the system considers a range of geometric features sufficient to give process capability guidelines. Sound process attribution and optimisation can only be achieved with the inclusion of further component data. This includes component dimensions, feature dimensions, critical tolerances, material type, sheet gauge and surface finish. Component and processes data, such as batch size, lead time, machine limitations, tool costs, machining costs and process availability must also be incorporated into the system. This will ensure that the processes selected for any component are capable and ensure cost effective manufacture.

Acknowledgements.

The research work carried out towards this paper has been funded by SERC.

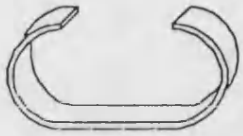
References.

1. Opitz,H. A classification system to describe workpieces. Pergamon Press. 1970.
2. Bedworth,D.D., Henderson,M.R. & Wolfe,P.M. Computer-Integrated Design And Manufacture. McGraw-Hill inc. 1991. pp 178-9.
3. Kimpel,J.F. & Richards,L.G. A Feature-based Group Technology Coding System for Sheet Metal Parts. Journal of Engineering Design. Vol.2, No 3, 1991.
4. Hoffmann,M., Geiβler,U. & Geiger,M. Computer-aided generation of bending sequences for die-bending machines. Journal of Materials Processing Technology, 30 (1992) 1-12.

Figure 1. The system code for figure 4 to demonstrate the code structure.

1 0	0 0	1 0 1 0 0	0 0 0 0 0 0
Perimeter Cutting Section.	Internal Cutting Section.	Folding Section.	Forming Section.

Figure 2. An example of a single axis enclosed fold. No enclosed features can be press formed as the sheet wraps round on itself. An enclosed rotationally symmetrical feature would have a similar interference feature. This would appear as a narrowing of the component's radius moving away from its base.



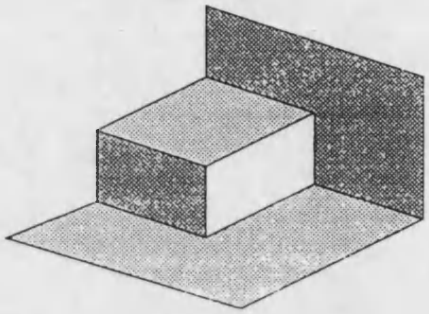


Figure 3. Simultaneous folds.

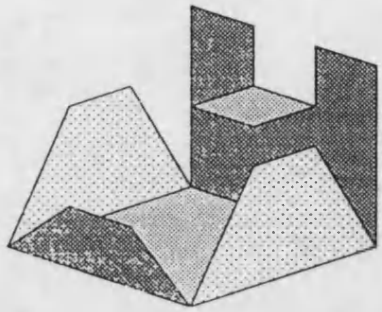


Figure 4. Folds with axial interference.

Table 1. Cutting perimeter catagories.

Entry	Feature Type.	Process Implications.
0 1	Straight edges with no internal angles.	This type can be cut using a shear or guillotine of any length.
1 0	Straight edges with internal angles or notches.	This type could be cut using standard notching and nibbling tools.
1 1	Perimeter involves complicated contours.	This type can only be cut using a CNC flame cutter or complex dedicated press tools.

Table 2. Cutting internal catagories.

Entry	Feature Type.	Process Implications.
0 0	No internal features.	
0 1	All features are common geometric shapes.	These can all be produced by punching and nibbling using standard tools.
1 0	Features present which include complicated contours.	These can only be produced using a CNC flame cutter or purpose made dedicated tooling

Table 3. Folding catagories.

Digit	Feature Type.	Process Implications.
1	Are any folds present?	No folding required.
2	Are there any enclosed folds? This occurs when there are three or more parallel folds in the same direction.	Although this factor does not guarantee that there will be interference problems in producing the folds, less than three such folds guarantees that there will not be interference problems (granting that any method of folding may be used).
3	Any folds with axial interference ? (see fig. 4.).	Interference at both ends of a fold, due to fold ends not lying on the component perimeter or other fold axes perpendicular to the fold in question, requires exact length folding tools.
4	Are any of the folds other than 90° ?	This requires that special tooling be used.
5	Any simultaneous folds present ? (see fig. 3.).	These need to be formed at the same time and thus require dedicated tools.

Table 4. Forming catagories.

Digit	Feature Type.	Process Implications	Examples.
1	Are any open single axis features present? i.e. the feature is uniform in one plane.	The only features which can be produced by simple rolling, but may be manufactured by other processes.	Channels, arced panels.
2	Are any enclosed single axis features present? i.e. deformation > 180° in one axis (see fig.2.).	Often require special tools i.e. open sided rollers, wrap forming tools or curling tools.	Hoops, uniform aerofoils sections.
3	Are any open rotationally symmetrical features present? i.e. rotationally symmetrical perpendicular to the sheet.	These are the only feature which can be flow formed, but it may also be manufactured by other processes.	Pan bodies.
4	Are any enclosed rotationally symmetrical features present? i.e. deformation > 180°.	This feature can be produced as catagory 3 and then post processed.	Beer can body.
5	Are any open unsymmetrical feature present?	These are generally pressed in large volume manufacture or hand beaten, explosive formed or peen formed in low volume manufacture.	Car or aircraft body panels.
6	Are any enclosed unsymmetrical features present? As type 5 but with a net deformation > 180°.	These components cannot be pressed to finished shape and may require multi-stage forming.	Enclosed body panels.

Automated Nesting of Sheet Metal Components onto Stock Sheets

A J Scott, A R Mileham

Manufacturing Group, School of Mechanical Eng, University of Bath, Claverton Down, Bath BA2 7AY

Abstract.

This paper describes the structure and operation of an automated system which can nest a large number of different sheet metal components onto stock sheets of material. The system is based on two heuristics, used sequentially, which can rapidly nest rectangular shapes of varying size and shape. Irregularly shaped components must be represented in this format prior to nesting, which compromises the potential performance of the solution, but allows simple data manipulation. The methods of enclosing components in rectangular envelopes are not dealt with in this work. When tested manually the heuristics perform well and they are currently being built into a computer system that will enable automated nesting and more comprehensive testing.

1. Introduction.

In the sheet metal environment nesting is constrained to locating two dimensional shapes on standard sizes of strip or sheet material. Further constraints may also be imposed such as components which must be grouped together or components which have pre-defined orientations. In a traditional production environment nesting patterns are limited to those formed by guillotine cuts, which span the width of the sheet being cut. The development of rectangular shears and laser cutting removes this constraint from many production environments.

A large amount of component design is now carried out using CAD or Solid Modelling and many of the sheet metal processes are now Numerically Controlled. Thus nesting is a favourable area of sheet metal planning to automate as it integrates these two areas. Currently, no automated system can mimic the highly developed spatial awareness and strong powers of geometric reasoning of a human planner. However, the addition of a perimeter cutting allowance, required for most cutting processes, and the evaluation of layout efficiency can be carried out more quickly by an automated system.

Nesting of rectangles of varying size and shape can be classified as a 2 dimensional cutting stock problem. Optimising the cutting of bar stock, a 1 dimensional problem, has been successfully tackled using Linear Programming techniques (Gilmore and Gomory 1961). This approach has been adapted, with limitations, to the 2 dimensional situation for rectangular components (Gilmore and Gomory 1965). The dynamic programming technique has also been applied to the problem of nesting rectangular components of varying size and shape (Haims and Freeman 1970). However, the performance of all of these systems is limited by the considerable constraints which have to be applied to allow their use, prompting other approaches to be considered.

One approach is to position components sequentially as close to the bottom left corner of the sheet as possible, this algorithm was developed to place rectangular components into bins with the aim of minimising the resulting height occupied (Baker et al 1980). However, no satisfactory method of optimising the sequence in which the components are placed, which is critical for efficient nesting, has been developed. The more sophisticated Next-Fit Decreasing-Height and First-Fit Decreasing-Height algorithms (Coffman et al 1980) perform better. These take components in height order and place them in rows. The First-Fit Decreasing-Height algorithm also checks if any component could be placed on the end of a row formed earlier. Israni and Sanders 1985 developed an algorithm which places rectangular components along the base and the left side of the sheet, with the components sequenced in length or height order. This algorithm was developed with the intention of subsequent human intervention. Adamowicz and Albano 1976(a) recognised that planners in industry frequently form strips of identical components, which allows easier manipulation. Their system carries this out automatically, places largest strips first and attempts to places smaller strips in the gaps of an existing layout.

There are also a number of systems which do not simplify components into rectangles, but nests their true shape. Qu and Sanders 1987 developed a system which places components into a rectangular construct envelope. If a large number of rectangles are permitted a highly accurate representation of even the most complicated components can be achieved; however this is at the expense of computational complexity. There are also a number of systems which deal with the actual shape or a low level of simplification (Adamowicz and Albano 1976(b), Dagli and Tatoglu 1987 and Tiow et al 1991). Due to the complexity of the data required for each component, the infinite range of positions and orientations, and the impossibility of predicting a good sequence these systems are slow and unsuitable nesting for a large number of components on the same sheet.

2. A Novel Nesting System.

Some existing systems carry out an exhaustive trial of all nesting sequences, assuring the optimum is selected. These are impractical for a large BOM, for instance 20 fixed orientation components have $2.433\ 10^{18}$ possible nesting sequences. This number is vastly increased if more than one component orientation is possible. Some systems take the components in a structured order, such as by decreasing height, but this is far from optimal and can only assure that worst case performance does not occur. However, this arbitrary ordering is quick, allowing computational time to be applied to component positioning. It would seem advantageous to develop a system which selects a component or group of components to match a space on the sheet. Rectangular envelope representation requires only two

dimensions, the side lengths, to be stored and only the two principal orientations to be considered. The methods of how components can be enclosed by a rectangular envelope will not be dealt with in this paper. Two new heuristics have been developed which, used sequentially, provide a layout which conforms to the guillotine cut constraint.

3. Strip forming Dimension Search (SFDS).

The heuristic operates by selecting a combination of component envelopes from the BOM (Bill of Materials) which can be positioned on the sheet to form a neat strip. To keep waste low the envelopes must be of a similar width and the sum of their lengths must be close to that of the sheet side. This is done in two stages. Initially all envelopes with one dimension within a pre-determined range are collected into a sub-group. The upper bound of this range is the strip width and the lower bound is the waste acceptance limit. The components' other dimensions are then summed in various combinations, with the object of finding a combination matching the required strip length. An acceptance range is also set for the required strip length ie. the sheet width. Any combination of envelopes found with lengths summing within the range are removed from the sub-group and nested. Diagram 1 shows the acceptance tolerances for the length and width of the strips. The upper width limit is the start of the next strip and the lower limit is shown by the broken line. The lower limit for strip length is also shown as a broken line and the upper limit is the width of the sheet.

A WIDTH ORIENTATED SFDS LAYOUT.

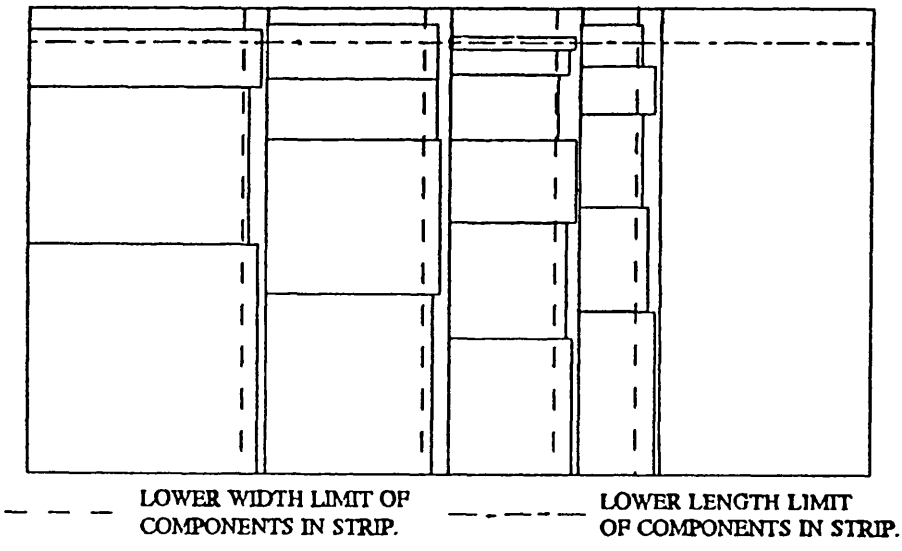


DIAGRAM 1.

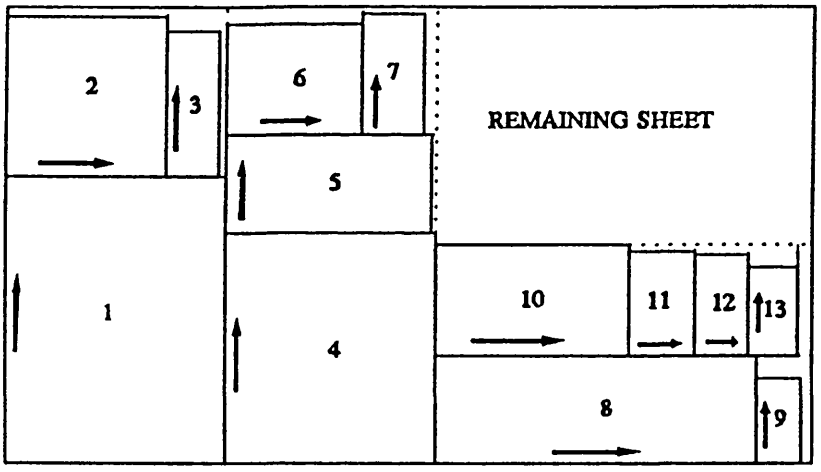
The strip combinations start with the largest envelopes and work towards the smaller envelopes. This allows the first strip found which meets the length requirements to be accepted. It is preferential to nest large components early as they are more awkward to use as pattern 'fill in'. The system then checks the remaining envelopes in the sub-group for other satisfactory combinations. If none are found those remaining are returned to the BOM, a new strip width range is set and envelopes with a dimension within this range are searched for. Envelopes not utilised to form a strip with the approximate width of their larger dimension

will be evaluated a second time for a strip with the approximate width of their smaller dimension. The exceptions to this are if the enclosed component is specified as having a fixed orientation, or if the envelope is a square. If the strip length and width tolerances are kept tight an efficient layout is produced. However, due to the nature of this heuristic's operation not all the components in the BOM will be nested. If the length and width tolerances are relaxed and this heuristic is used again the remaining components can be nested, however the layout produced would be very inefficient. Alternatively a different heuristic could be employed to nest these remaining component envelopes.

4. Dimension Fitting Area Search (DFAS).

This heuristic is far more suited to smaller BOMs containing great variations in component size and shape, as will remain after the SFDS heuristic has been used on a large BOM. It operates by identifying the largest component envelope in the BOM which will fit into a space remaining on the sheet. These spaces are maintained in the form of rectangles, and their areas are recorded as well as their dimensions. The envelopes to be nested are ordered by decreasing area. Diagram 2 shows a layout nested by this system.

SQUARE REDUCING AFDS LAYOUT.



NUMBERS INDICATE ORDER OF COMPONENT NESTING.
ARROWS SHOW THE DIRECTION IN WHICH THE SHEET IS FILLED.

DIAGRAM 2.

The system commences its search at the first component envelope of equal or less area than the available space. The dimensions of the component's envelope are then checked to ensure it will fit into the space. As the envelopes are in order of size the first match found will be the best possible. Any area remaining from this process will be used for further nesting, providing suitable components exist in the BOM. If the space is too large i.e. no envelope shares a dimension with it, the largest component envelope in the BOM will be located in the bottom left of the space. From the corner of this envelope either a horizontal or vertical line can be drawn to divide the remaining area into two rectangles. The line which gives the smaller area will be used. This is the primary area to nest. Thus the largest rectangular area possible is kept unused at any one time.

5. Conclusion.

This paper has investigated the automated nesting of sheet metal parts onto prescribed sizes of stock sheet. A review of the existing nesting systems has been carried out and their limitations identified. Two new heuristics have been developed which sequence, as well as position, rectangular envelopes. These differ from existing heuristics as they choose the sequence of components to suit the sheet size, rather than optimise the position and orientation of components taken in an arbitrary sequence. The heuristics have the potential to be highly effective in the nesting of sheet metal parts.

6. Further Work.

Programs to execute the heuristics are currently being developed using the Microsoft C 5.1 system. A method has been developed to improve the layouts generated by removing the guillotine cut constraint which will be incorporated into the system. A range of BOMs containing different numbers of components, average component size and component aspect ratio (length to width ratio) will be tested. This will highlight the strengths and weaknesses of the heuristics and the most suitable environments for their application.

7. Acknowledgements.

The research work carried out towards this paper has been funded by SERC.

8. References.

Adamowicz M. and Albano A. 1976(a). A solution of the rectangular cutting-stock problem. IEEE Trans. Syst. Man. And Cybernetics, Vol. SMC-6, No. 4, April.

Adamowicz M. and Albano A. 1976(b). Nesting two-dimensional shapes in rectangular modules. Computer Aided Design. Vol. 8, No. 1, Jan.

Baker B.S., Coffman E.G.Jr. and Rivest R.L. 1980. Orthogonal packing in two dimensions. SIAM J. COMPUT. Vol. 9, No. 4, Nov.

Coffman E.G.Jr., Garey M.R., Johnson D.S. and Tarjan R.E. 1980. Performance bounds for level-oriented two-dimensional packing algorithms. SIAM J. COMPUT. Vol. 9, No. 4, Nov.

Dagli C.H. and M. Tatoglu M.Y. 1987. An approach to two-dimensional cutting stock problems. Vol. 25, No. 2, pp 175-190.

Gilmore P.C. and Gomory R.E. 1961. A Linear Programming approach to the Cutting Stock Problem. Opns. Res., 9, 849-859.

Gilmore P.C. and Gomory R.E. 1965. Multi-stage Cutting Stock Problems in two or more dimensions. Opns. Res., 13, 94-128.

Haims J.M. and Freeman H. 1970. A multistage solution of the template-layout problem. IEEE Trans. Syst. Sci. and Cybernetics, Vol. SSC-6, No. 2, April.

Israni S.S. and Sanders J.L. 1985. Performance testing of rectangular parts-nesting heuristics. Int. J. Prod. Res. Vol. 23, No. 3, pp 437-456.

Qu W. and Sanders J.L. 1987. A nesting algorithm for irregular parts and factors affecting trim losses. Int. J. Prod. Res. Vol. 25, No. 3, pp 381-397.

Tiow O.K., Wah P.W. and Guang W.C. 1992. Optimisation of materials processing using AutoCAD. J. Mats. Proc. Tech, 29, pp 293-2999.

A new system to nest sheet metal parts which have been enclosed within rectangular envelopes.

A. J. Scott.
Dr. A. R. Mileham.

Abstract.

An automated nesting system for sheet metal components is described which is being developed to meet the demands of the low volume, high variety production environment. The system relies on individual components being placed within rectangular envelopes to simplify their subsequent nesting on to stock sheets. As there are a number of established methods which can effectively achieve this for one component, or a cluster of components, this aspect has not been considered. This work concentrates on the subsequent part of the system, which generates layouts of the enveloped components on specified stock sheets. This part of the system consists of three heuristics; the first creates strips of components, the second improves these strips and the third nests any components not yet dealt with. A fully automated version of this system has been developed in the C language and benchmarked against two established nesting heuristics. Some details of the system's performance are discussed.

Introduction.

In the sheet metal environment nesting is constrained to locating two dimensional shapes on standard sizes of strip or sheet material. Further constraints may also be imposed such as components which must be grouped together or components which have pre-defined orientations. In a traditional production environment nesting patterns are limited to those formed by guillotine cuts, which span the width of the sheet being cut. The development of rectangular shears and laser cutting removes this constraint from many production environments.

A large amount of component design is now carried out using CAD or Solid Modelling and many of the sheet metal processes are now Numerically Controlled. Thus nesting is a favourable area of sheet metal planning to automate as it integrates these two areas. Currently, no automated system can mimic the highly developed spatial awareness and strong powers of geometric reasoning of a human planner. However, the speed of a computer allows either exhaustive layout systems to be used for small parts list's or fast non-optimal systems to be developed. Also, tasks such as the addition of a bridge gap (perimeter cutting allowance), required for most cutting processes, and the evaluation of the efficiency of the layout can be carried out more quickly by an automated system.

Approaches to Nesting.

Nesting of rectangles of varying size and shape can be classified as a 2 dimensional cutting stock problem. Optimising the cutting of bar stock, a 1 dimensional problem, has been successfully tackled using Linear Programming techniques (Gilmore and Gomory[1]). This

approach has been adapted, with limitations, to the 2 dimensional situation for rectangular components (Gilmore and Gomory[2]). The dynamic programming technique has also been applied to the problem of nesting rectangular components of varying size and shape (Haims and Freeman[3]). However, the performance of all of these systems is limited by the considerable constraints which have to be applied to allow their use, prompting other approaches to be considered.

One approach is to position components sequentially as close to the bottom left corner of the sheet as possible, this algorithm was developed to place rectangular components into bins with the aim of minimising the resulting height occupied (Baker et al[4]). However, no satisfactory method of optimising the sequence in which the components are placed, which is critical for efficient nesting, has been developed. The more sophisticated Next-Fit Decreasing-Height and First-Fit Decreasing-Height algorithms (Coffman et al[5]) perform better. These take components in height order and place them in rows. The First-Fit Decreasing-Height algorithm also checks if any component could be placed on the end of a row formed earlier. Both these algorithms are used as benchmarks for the evaluation of the new algorithms developed. Israni and Sanders[6] developed an algorithm which places rectangular components along the base and the left side of the sheet, with the components sequenced in length or height order. This algorithm was developed with the intention of subsequent human intervention. Adamowicz and Albano[7] recognised that planners in industry frequently form strips of identical components, which allows easier manipulation. Their automated system places largest strips first and then attempts to place smaller strips in the gaps of an existing layout. However it is limited by only using identical parts to form strips.

There are also a number of systems which do not simplify components into rectangles, but nests their true shape. Qu and Sanders[8] developed a system which places components into a rectangular construct envelope. If a large number of rectangles are permitted a highly accurate representation of even the most complicated components can be achieved; however this is at the expense of computational complexity. There are also a number of systems which deal with the actual shape or a low level of simplification (Adamowicz and Albano[9], Dagli and Tatoglu[10] and Tiow et al[11]). Due to the complexity of the data required for each component, the infinite range of positions and orientations, and the impossibility of predicting a good sequence these systems are slow and unsuitable nesting of a large number of components onto the same sheet.

A Novel Nesting System.

Some existing systems carry out an exhaustive trial of all nesting sequences, assuring the optimum is selected. These are impractical for a large parts lists, for instance 20 fixed orientation components have $2.433\ 10^{18}$ possible nesting sequences. This number is vastly increased if more than one component orientation is possible. Some systems take the components in a structured order, such as by decreasing height, but this is far from optimal and can only assure that worst case performance does not occur. However, this arbitrary ordering is quick, allowing computational time to be applied to component positioning. It would seem advantageous to develop a system which selects a component or group of components to match a space on the sheet. Rectangular envelope representation requires only the side lengths to be stored and only the two principal orientations to be considered.

A nesting system based on this rectangular format has been developed and its general structure is shown in Diagram 1. In the diagram two elements of the system are enclosed within broken line boxes; these are concerned with the placement of components within rectangular envelopes. Methods exist to do this (Qu and Sanders[8]) and a new method to efficiently cluster identical components into rectangles is being developed. However this aspect of the system is beyond the scope of this paper.

The core of the system is the 'Strip Forming Dimension Search' algorithm and the 'Area Fitting Dimension Search' algorithm (Scott and Mileham[12]). The former creates efficient strips of parts, but can rarely nest all parts, and the latter places any remaining components. The third element of the system, the 'Strip Squeeze' algorithm, improves the strip layout which has been generated. The system as shown will produce a part layout which does not conform to the 'guillotine cut constraint'. However, this constraint is adhered to if the 'Strip Squeeze' algorithm is not used, allowing the operator to choose a constrained layout if it is required. The operation of these algorithms is explained in detail later.

The rectangular envelope format allows the nesting system to be easily interfaced to a CAD input as limited information is required. The system outputs the co-ordinate position of the component's envelope which can be passed to an automated CAM system to allow planing of the cutting process.

In the automated nesting system developed, the rectangular envelopes to be nested are read in from data files which hold parts lists designed to have particular features. Additional data such as the stock sheet size and bridge gap to be used are entered from the keyboard. After the system has nested the parts, two output files are created. The results file contains the sheet number, position and orientation of each component. The analysis file holds the statistical output such as the nesting efficiency achieved and the time taken.

Flow Diagram of the Total Nesting System.

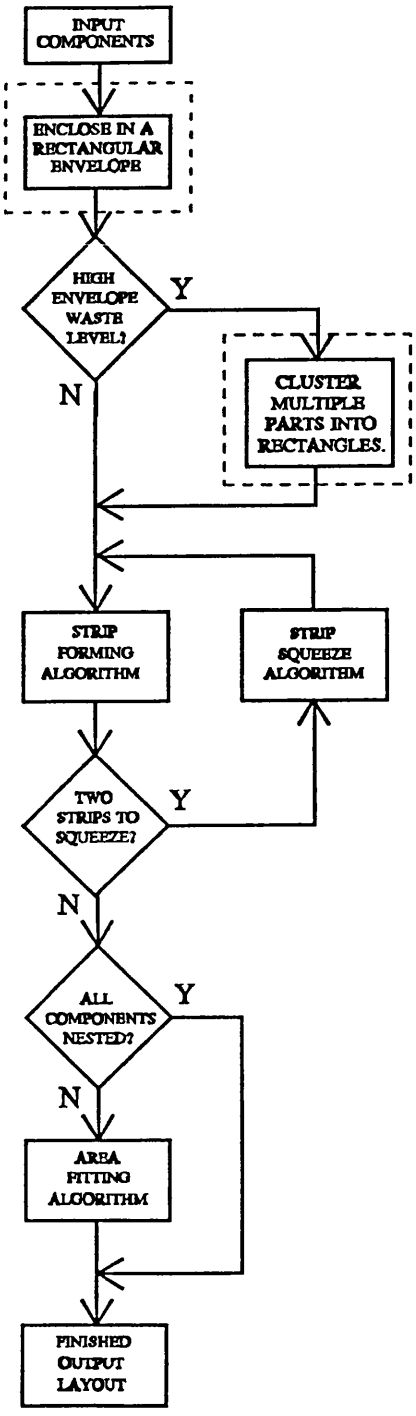


Diagram 1.

Strip Forming Dimension Search.

This heuristic operates by selecting a combination of component envelopes from the parts list (Bill of Materials) which can be positioned on the sheet to form a neat strip. To keep waste low the envelopes must be of a similar width and the sum of their lengths must be close to that of the sheet side. This is done in two stages. Initially all envelopes with one dimension within a pre-determined range are collected into a sub-group. The upper bound of this range is the largest component which sets the strip width. The lower bound is the product of the strip width and the waste acceptance limit. The components' other dimensions are then summed in various combinations, with the object of finding a combination matching the required strip length. An acceptance range is also set for the required strip length ie. the sheet width. Any combination of envelopes found with lengths summing within the range are removed from the sub-group and nested. Diagram 2 shows the step by step process of creating a strip of parts.

The Operation of the 'Strip Forming Dimension Search' Algorithm.

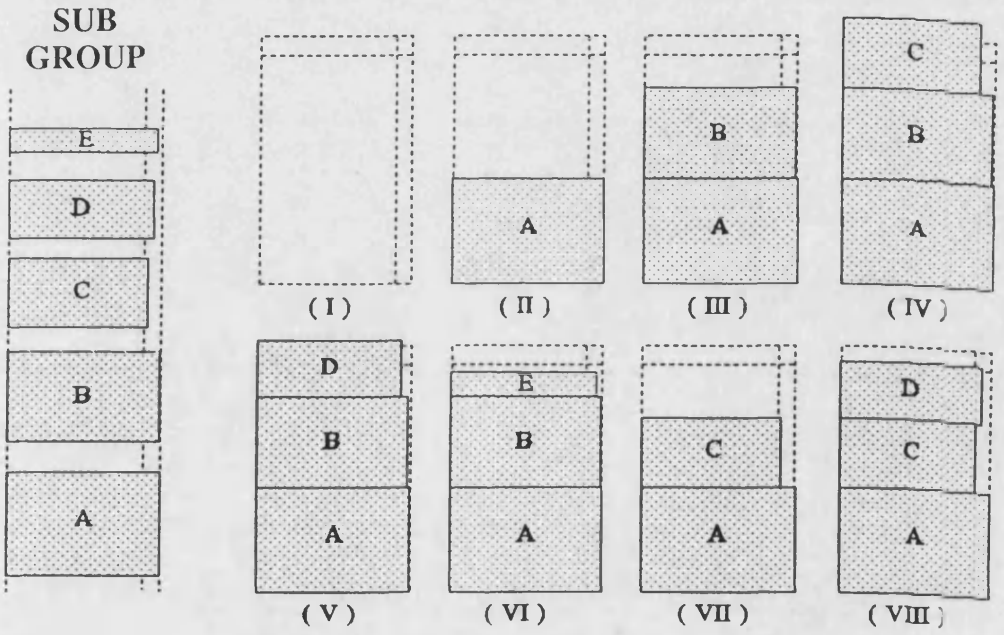


Diagram 2.

All Envelopes with a dimension within the strip width tolerance are removed from the main parts list and placed in a sub group in order of decreasing height, as shown above (working from the bottom up).

(I) Shows the length and width tolerances for the strip.

(II) and (III) Envelopes A and B are placed in the strip and, in each case, it is checked that they do not exceed or meet the strip length tolerance.

(IV),(V) and (VI) Remaining envelopes C, D and E are all tried, but none meet the strip length tolerance.

(VII) To allow further envelope combinations to be tried, envelope B is removed and replaced by envelope C. There is no need to check the strip length tolerance as the height ordering of the sub group assures that B is not bigger than C.
(VIII) Envelope D is added and a combination which meets the strip length tolerances is found. If there are other combinations which meet the tolerance they would not include as many large components, however they may provide a more efficient layout. This is accepted to avoid the need for exhaustive trials of all envelope combinations.

Diagram 3 shows a completed 'Strip Forming Dimension Search' layout. The broken lines represent the acceptance tolerances for the length and width of the strips. The system tries to form wide strips first in order to place the larger components as soon as possible. Within each strip the combinations start with the largest Y dimensions envelopes and work towards the envelopes with smaller Y dimensions. This aims to nest the largest components first by assuming that all the X dimensions are approximately the same. It also allows the first strip found meeting the length requirements to be accepted. It is preferential to nest large components early as they are more awkward to use as pattern 'fill in'. The system then checks the remaining envelopes in the sub-group for other satisfactory combinations. If none are found those remaining are returned to the parts list, a new strip width range is set and envelopes with a dimension within this range are searched for.

A Layout created by the 'Strip Forming Dimension Search' Algorithm.

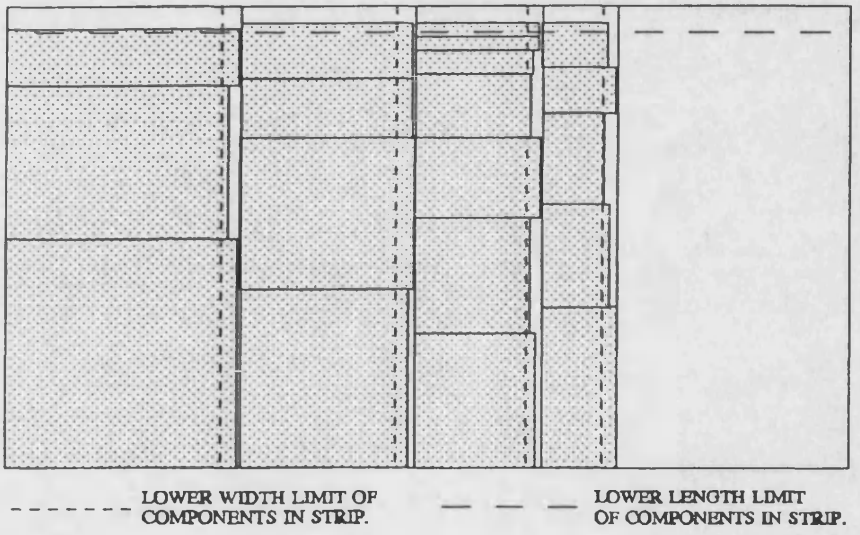


Diagram 3.

Envelopes not utilised to form a strip with the approximate width of their larger dimension will be evaluated a second time for a strip with the approximate width of their smaller dimension. The exceptions to this are if the enclosed component is specified as having a fixed orientation, or if the envelope is a square. If this system is used with tight strip length and width tolerances an efficient layout is produced. However, due to the nature of this heuristic's operation not all the components in the parts list will be nested. If the length and width

tolerances are relaxed and this heuristic is used again the remaining components can be nested, however the layout produced would be very inefficient. Alternatively a different heuristic could be employed to nest these remaining component envelopes.

The Strip Squeezing Algorithm.

The 'Strip Forming' algorithm generates a layout which conforms to the guillotine cut constraint i.e. the parts can be cut from the sheet using straight line cuts across the sheet's span. This constraint is often not required with modern cutting methods (laser and rectangular shears) and the layout efficiency can only be improved by its removal. Rather than developing a completely new algorithm to generate an unconstrained layout, an improvement algorithm could be applied to the existing layout. Such an algorithm has been developed and it enhances the 'Strip Forming' algorithm's layout by squeezing pairs of strips together. It should be noted that no additional improvement in this section of the system will be seen when component rotation is permitted as the orientations are inherited from the 'Strip Forming' algorithm. The detailled operation of this algorithm is shown below in diagram 4.

The Stages of the 'Strip Squeezing' Algorithm.

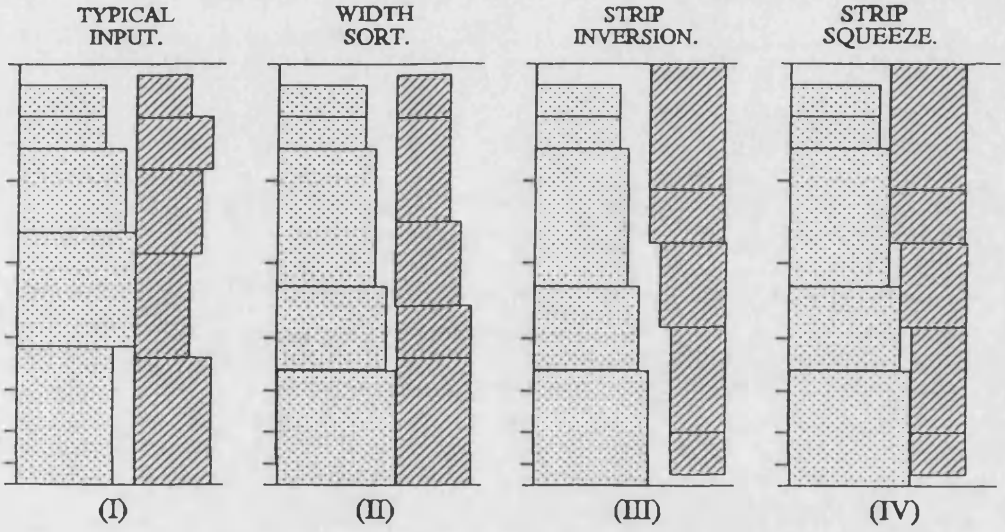


Diagram 4.

(I) Shows two strips conforming to the 'guillotine cut constraint' which have been generated by the 'Strip Forming' algorithm; this is the layout which the 'Strip Squeezing' Algorithm receives. The parts are in decreasing Y dimension order, with the largest Y dimensions at the bases of the strips.
(II) The parts making up each strip have been re-arranged into decreasing X dimension order, again working from the bases of the strips.
(III) The second operation of this algorithm is to invert the strip and align the parts to the right, this corresponds geometrically to a 180° rotation.

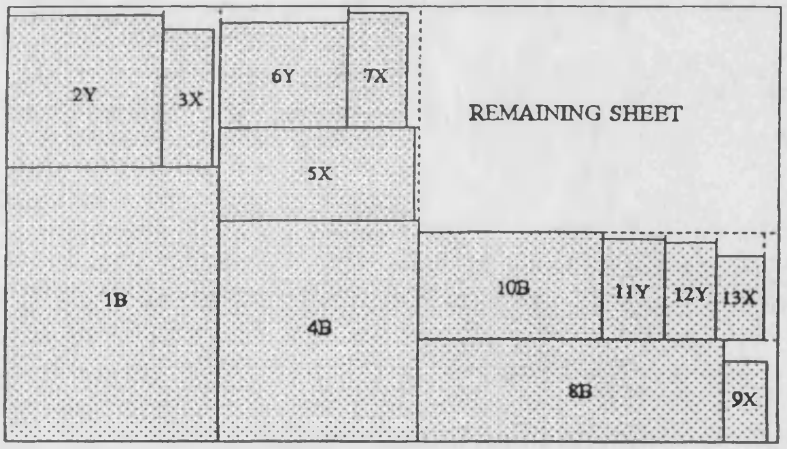
(IV) Finally the strips can be squeezed together, reducing the area of sheet which they occupy. The smallest gap between the strips is calculated. This is the maximum translation of the right strip which can be performed without part overlap occurring.

Area Fitting Dimension Search.

This heuristic is far more suited to smaller parts lists containing great variations in component size and shape, as will remain after the 'Strip Forming' algorithm has been used on a large parts list. It operates by identifying the largest component envelope in the parts list which will fit into a space remaining on the sheet. These spaces are maintained in the form of rectangles, and their areas are recorded as well as their dimensions. The envelopes to be nested are ordered by decreasing area.

The system commences its search at the first component envelope of equal or less area than the available space. The dimensions of the component's envelope are then checked to ensure it will fit into the space. As the envelopes are in order of size the first match found will be the best possible. Any area remaining from this process will be used for further nesting, providing suitable components exist in the parts list. If the space is too large i.e. no envelope shares a dimension with it, the largest component envelope in the parts list will be located in the bottom left of the space. From the corner of this envelope either a horizontal or vertical line can be drawn to divide the remaining area into two rectangles. The line which gives the smaller area will be used. This is the primary area to nest. Thus the largest rectangular area possible is kept unused at any one time. A step by step illustration of this algorithm's operation is given in Diagram 6 and a full sheet layout is shown in diagram 5.

A Layout created by the 'Area Fitting Dimension Search' Algorithm.



NUMBERS INDICATE ORDER OF COMPONENT NESTING,
LETTERS INDICATE HOW THE COMPONENT WAS PLACED.
B = BOTTOM LEFT CONTINGENCY. X = X DIM. FIT. Y = Y DIM. FIT.

Diagram 5.

The Operation of the 'Area Fitting Dimension Search' Algorithm.

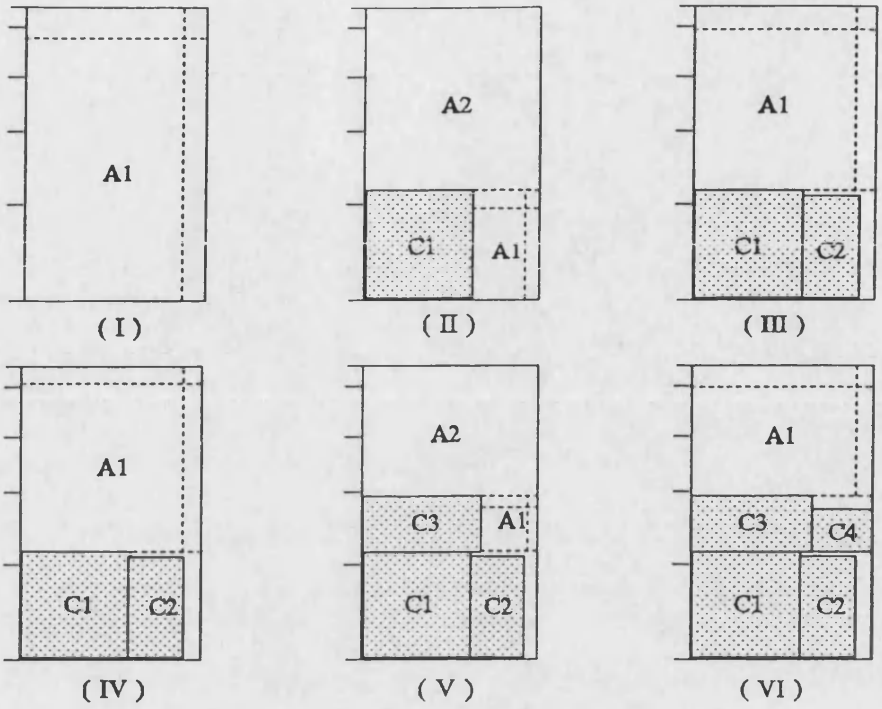


Diagram 6.

(I) Shows the remaining area at the end of a sheet after the Strip Forming heuristic has been used. The dotted lines show the size tolerances, one of which a component must meet to be initially placed in the space.
(II) If no component is large enough to meet the tolerances, as in this case, a contingency rule is used. This is to place the largest envelope from the parts list , C1, in the bottom left corner of the space. The remaining area is now divided into A1 (the primary area to nest) and A2 (the remaining area). The area could have been divided using a vertical line from the top right corner of C1. However, it is considered better to maximise the size of the 'largest remaining area' at all times.
(III) Envelope C2 fits A1.
(IV) No envelope fits the new A1.
(V) Largest remaining envelope C3 is placed creating new spaces A1 and A2.
(VI) Envelope C4 fits A1.

Note: A1 and A2 in this example represent the first two positions in a 'stack' of areas to be filled, which is why their values change. In this example the stack is never loaded with more than two areas.

Testing of the system.

Isolated testing of the new system would be meaningless as the general materials efficiencies reported in industry are not for purely rectangular parts and they vary greatly. Direct comparison with human planners or commercial systems would require the rectangular enveloping 'front end' to be developed. Therefore, at this stage the most meaningful test is to compare the new system against the existing 'intervention free' algorithms for rectangular parts. The two algorithms to be used as benchmarks are the 'Next-Fit Decreasing-Height' and 'First-Fit Decreasing-Height' algorithms (Coffman[5]) which are described above. As both of these algorithms are suitable for improvement with the 'Row Squeezing' algorithm these combinations are to be tested. Both the 'Strip Forming Dimension Search' and 'Area Fitting Dimension Search' algorithms developed can be used individually and are also to be tested.

In order to evaluate the efficiency of a layout it is necessary to define what will be considered as waste. With completely nested sheets there is rarely any doubt that all the unused sheet is waste, although occasionally a particularly large area may remain which would be worth retaining for future use. On part filled sheets, defining what of the remaining area can be considered useful stock material is more difficult. One method is to take the area which the algorithm could have continued to use, had the parts list been larger. However, for simple stock management, only large rectangular sections would be retained for future use, so the above method of evaluation may not always reflect practical material savings. Insufficient tests have been carried out to give the average performance of each algorithm accurately. The maximum and minimum percentage nesting efficiencies found so far for each algorithm are given in Table 1. In addition to performing better, the new algorithms are also more consistent.

The absolute performance of a nesting algorithm is hard to define as some waste, such as the 'bridge gap', cannot be avoided. One approach is to create a parts list with a known benchmark layout which cannot be improved. The layout structure must not allow any test algorithm to duplicate it. The layout used (Diagram 7) has 25 parts, separated by a 0.02m bridge gap, occupying 60% of a 2.5m X 1.5m sheet. Layouts generated by the benchmark algorithms (Diagrams 8 & 9), the 'Strip Forming' and 'Strip Squeezing' algorithms (Diagram 10), the 'Area Fitting' algorithm (Diagram 11) and the total system (Diagram 12) are shown.

Conclusion.

This paper has examined the nesting of sheet metal parts onto prescribed sizes of stock sheet. A review of existing nesting systems has been carried out and their limitations have been identified. A new nesting system, based on enclosing parts within rectangular envelopes, has been outlined. The placing of the rectangular envelopes is carried out by three novel algorithms, used sequentially, which have been described in detail. Many existing algorithms place the components in either an arbitrary or a pre-determined sequence. In contrast the algorithms introduced use rules to choose the best component to place at each stage. Thus the sequence of component placement is indirectly evolved during the nesting process. The initial tests give cause for optimism. The new algorithms have not been out performed by the existing algorithms for any test parts lists tried. In addition, for many of the test parts lists the new system's results have been radically better than existing systems.

Table 1.

The Perfect Layout for the Part List.

Diagram 7.

The Next-Fit Decreasing-Height Layout.

Diagram 8.

The First-Fit Decreasing-Height Layout.

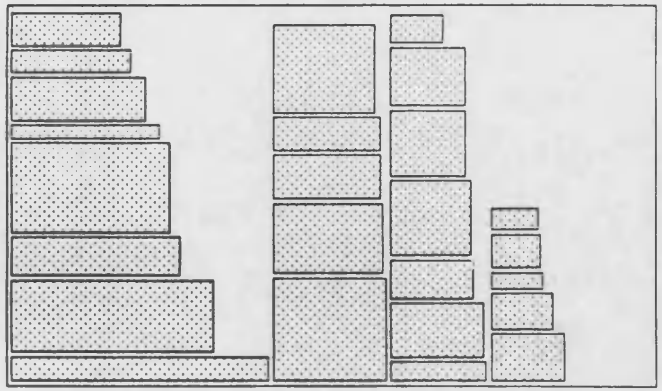


Diagram 9.

The Strip Forming And Squeezing Layout.

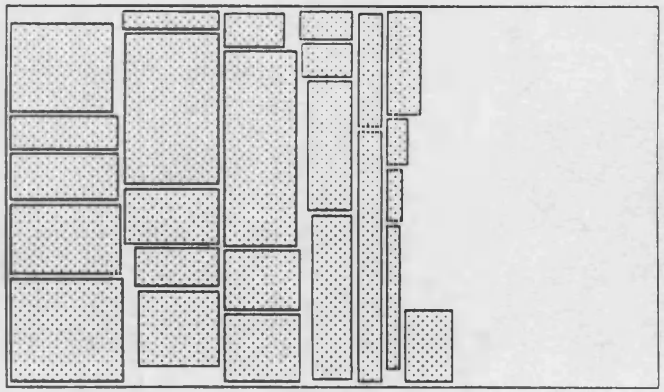


Diagram 10.

The Area Fitting Layout.

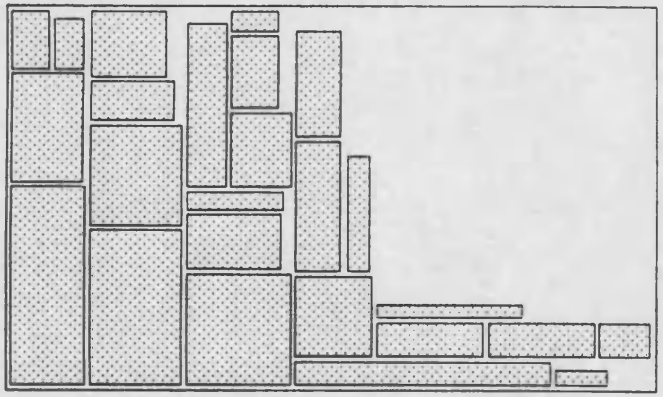


Diagram 11.

The Total System Layout.

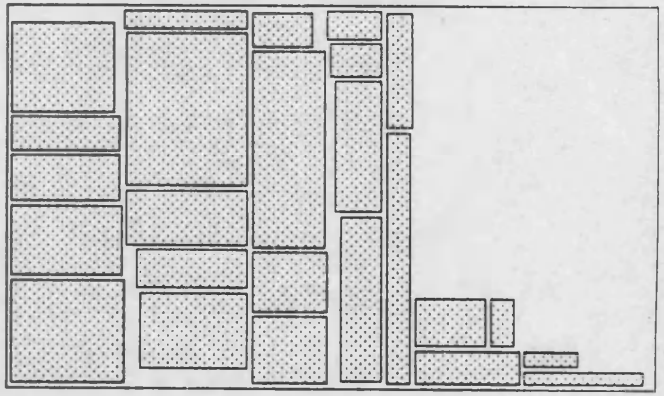


Diagram 12.

Further work.

A comprehensive set of structured tests are required to accurately evaluate the performance of the new algorithms against the established algorithms. The work of Israni and Sanders[6] has highlighted the effects on layout efficiency of the part aspect ratio (length / width) and the part size : sheet size ratio. In addition, the size of the parts list, the sheet aspect ratio and whether the parts may be rotated may also affect the nesting performance. The effects of these characteristics will be investigated to establish the robustness of each algorithm to them.

Both the 'Strip Forming Dimension Search' and 'Area Fitting Dimension Search' require part size acceptance limits to be set. These represent the greatest waste around a part which the algorithm should accept and still place components. With tight limits for the more efficient first part of the system, a large number of parts will be left for the less efficient latter parts of the system. If these acceptance limits are too loose the algorithms will not operate to their full potential. The optimum acceptance limit must be established by experimentation and may vary according to the characteristics of the parts to be nested and the stock sheet used.

Acknowledgements.

The research work carried out towards this paper has been funded by SERC.

References.

1. Gilmore P.C. and Gomory R.E. A Linear Programming approach to the Cutting Stock Problem. Opns. Res., 9, 849-859, 1961.
2. Gilmore P.C. and Gomory R.E. Multi-stage Cutting Stock Problems in two or more dimensions. Opns. Res., 13, 94-128, 1965.
3. Haims J.M. and Freeman H. A multistage solution of the template-layout problem. IEEE Trans. Syst. Sci. and Cybernetics, Vol. SSC-6, No. 2, April 1970.
4. Baker B.S., Coffman E.G.Jr. and Rivest R.L. Orthogonal packing in two dimensions. SIAM J. COMPUT. Vol. 9, No. 4, Nov 1980.
5. Coffman E.G.Jr., Garey M.R., Johnson D.S. and Tarjan R.E. Performance bounds for level-oriented two-dimensional packing algorithms. SIAM J. COMPUT. Vol. 9, No. 4, Nov 1980.
6. Israni S.S. and Sanders J.L. Performance testing of rectangular parts-nesting heuristics. Int. J. Prod. Res. Vol. 23, No. 3, pp 437-456, 1985.
7. Adamowicz M. and Albano A. A solution of the rectangular cutting-stock problem. IEEE Trans. Syst. Man. And Cybernetics, Vol. SMC-6, No. 4, April 1976.
8. Qu W. and Sanders J.L. A nesting algorithm for irregular parts and factors affecting trim losses. Int. J. Prod. Res. Vol. 25, No. 3, pp 381-397, 1987.
9. Adamowicz M. and Albano A. Nesting two-dimensional shapes in rectangular modules. Computer Aided Design. Vol. 8, No. 1, Jan 1976.
10. Dagli C.H. and M. Tatoglu M.Y. An approach to two-dimensional cutting stock problems. Vol. 25, No. 2, pp 175-190, 1987.
11. Tiow O.K., Wah P.W. and Guang W.C. Optimisation of materials processing using AutoCAD. J. Mats. Proc. Tech, 29, pp 293-2999, 1992.
12. Scott A.J. and Mileham A.R. Automated Nesting of Sheet Metal Components onto Stock Sheets. Advances in Manf. Tech. VII. pp 242-246. University of Bath, Sept. 1993.

A new system to nest sheet metal parts in the 'low volume, high variety' environment.

A. J. Scott.
Dr. A. R. Mileham.

Abstract.

A fully automated nesting system for sheet metal components is described which is being developed to meet the demands of the low volume, high variety production environment. The system relies on individual components being placed within rectangular envelopes to simplify their subsequent nesting on to stock sheets. As there are a number of established methods which can effectively achieve this for one component, or a cluster of components, this aspect has not been considered. This work concentrates on the subsequent part of the system, which generates layouts of the enveloped components on specified stock sheets. This part of the system consists of three heuristics; the first creates strips of components, the second improves these strips and the third nests any components not yet dealt with. A fully automated version of this system has been developed in the C language and benchmarked against two established nesting heuristics. Details of the system's performance are discussed.

Introduction.

In the sheet metal environment nesting is constrained to locating two dimensional shapes on standard sizes of strip or sheet material. Further constraints may also be imposed such as components which must be grouped together or components with fixed orientations. In a traditional production environment nesting patterns are limited to those formed by guillotine cuts, which span the width of the sheet being cut. The development of rectangular shears and laser cutting removes this constraint from many production environments.

A large amount of component design is now carried out using CAD or Solid Modelling and many of the sheet metal processes are now Numerically Controlled. Thus nesting is a favourable area of sheet metal planning to automate as it integrates these two areas. Currently, no automated system can mimic the highly developed spatial awareness and strong powers of geometric reasoning of a human planner. However, the speed of a computer allows either exhaustive layout systems to be used for small parts list's or fast non-optimal systems to be developed. Also, tasks such as the addition of a bridge gap (perimeter cutting allowance), required for most cutting processes, and the evaluation of the efficiency of the layout can be carried out more quickly by an automated system.

Approaches to Nesting.

Nesting of rectangles of varying size and shape can be classified as a 2 dimensional cutting stock problem. Optimising the cutting of bar stock, a 1 dimensional problem, has been successfully tackled using Linear Programming techniques (Gilmore and Gomory[1]). This approach has been adapted, with limitations, to the 2 dimensional situation for rectangular components (Gilmore and Gomory[2]). The dynamic programming technique has also been applied to the problem of nesting rectangular components of varying size and shape (Haims and Freeman[3]). However, the performance of these systems is limited by the considerable constraints which must be applied for their use, prompting other approaches to be considered.

One approach is to position components sequentially as close to the bottom left corner of the sheet as possible, this algorithm was developed to place rectangular components into bins with the aim of minimising the resulting height occupied (Baker et al[4]). However, no satisfactory method of optimising the sequence in which the components are placed, which is critical for efficient nesting, has been developed. The more sophisticated Next-Fit Decreasing-Height and First-Fit Decreasing-Height algorithms (Coffman et al[5]) perform better. These take components in height order and place them in rows. The First-Fit Decreasing-Height algorithm also checks if any component could be placed on the end of a row formed earlier. Both these algorithms are used as benchmarks for the evaluation of the

new algorithms developed. Israni and Sanders[6] developed an algorithm which places rectangular components along the base and the left side of the sheet, with the components sequenced in length or height order. This algorithm was developed with the intention of subsequent human intervention. Adamowicz and Albano[7] recognised that planners in industry frequently form strips of identical components, which allows easier manipulation. Their automated system places largest strips first and then attempts to place smaller strips in the gaps of an existing layout. However it is limited by only using identical parts to form strips.

There are also a number of systems which do not simplify components into rectangles, but nests their true shape. Qu and Sanders[8] developed a system which places components into a rectangular construct envelope. If a large number of rectangles are permitted a highly accurate representation of even the most complicated components can be achieved; however this is at the expense of computational complexity. There are also a number of systems which deal with the actual shape or a low level of simplification (Adamowicz and Albano[9], Dagli and Tatoglu[10] and Tlow et al[11]). Due to the complexity of the data required for each component, the infinite range of positions and orientations, and the impossibility of predicting a good sequence these systems are slow and unsuitable nesting of a large number of components onto the same sheet.

A Novel Nesting System.

Some existing systems carry out an exhaustive trial of all nesting sequences, assuring the optimum is selected. These are impractical for a large parts lists, for instance 20 fixed orientation components have $2.433\ 10^{18}$ possible nesting sequences. This number is vastly increased if more than one component orientation is possible. Some systems take the components in a structured order, such as by decreasing height, but this is far from optimal and can only assure that worst case performance does not occur. However, this arbitrary ordering is quick, allowing computational time to be applied to component positioning. It would seem advantageous to develop a system which selects a component or group of components to match a space on the sheet. Rectangular envelope representation requires only the side lengths to be stored and only the two principal orientations to be considered.

A nesting system based on this rectangular format has been developed and its general structure is shown in Diagram 1. In the diagram two elements of the system are enclosed within broken line boxes; these are concerned with the placement of components within rectangular envelopes. Methods exist to do this (Qu and Sanders[8]) and a new method to cluster identical components in rectangles is being developed. However this aspect of the system is beyond the scope of this paper.

Flow Diagram of the Total Nesting System.

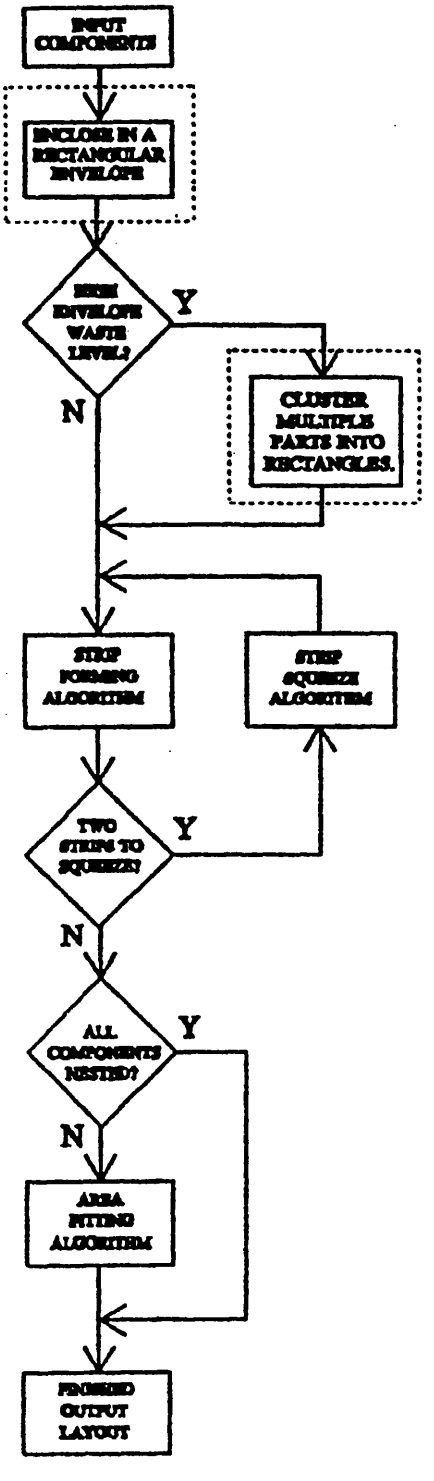


Diagram 1.

The core of the system is the 'Strip Forming Dimension Search' algorithm and the 'Area Fitting Dimension Search' algorithm (Scott and Mileham [13]). The former creates efficient strips of parts, but can rarely nest all parts, and the latter places any remaining components. The third element of the system, the 'Strip Squeeze' algorithm, improves the strip layout which has been generated. The system as shown will produce a part layout which does not conform to the 'guillotine cut constraint'. However, this constraint is adhered to if the 'Strip Squeeze' algorithm is not used, allowing the operator to choose a constrained layout if it is required. The operation of these algorithms is explained in detail later.

The rectangular envelope format allows the nesting system to be easily interfaced to a CAD input as limited information is required. The system outputs the co-ordinate position of the component's envelope which can be passed to an automated CAM system to allow planing of the cutting process. In the nesting system developed, the rectangular envelopes to be nested are read in from data files which hold parts lists designed to have particular features. Additional data such as the stock sheet size and bridge gap to be used are entered from the keyboard. After the system has nested the parts, two output files are created. The results file contains the sheet number, position and orientation of each component. The analysis file holds the statistical output such as the nesting efficiency achieved and the time taken.

Strip Forming Dimension Search.

This heuristic operates by selecting a combination of component envelopes from the parts list (Bill of Materials) which can be positioned on the sheet to form a neat strip. To keep waste low the envelopes must be of a similar width and the sum of their lengths must be close to that of the sheet side. This is done in two stages. Initially all envelopes with one dimension within a pre-determined range are collected into a sub-group. The upper bound of this range is the largest component which sets the strip width. The lower bound is the product of the strip width and the waste acceptance limit. The components' other dimensions are then summed in various combinations, with the object of finding a combination matching the required strip length. An acceptance range is also set for the required strip length ie. the sheet width. Any combination of envelopes found with lengths summing within the range are removed from the sub-group and nested. Diagram 2 shows the step by step process of creating a strip of parts.

The Operation of the 'Strip Forming Dimension Search' Algorithm.

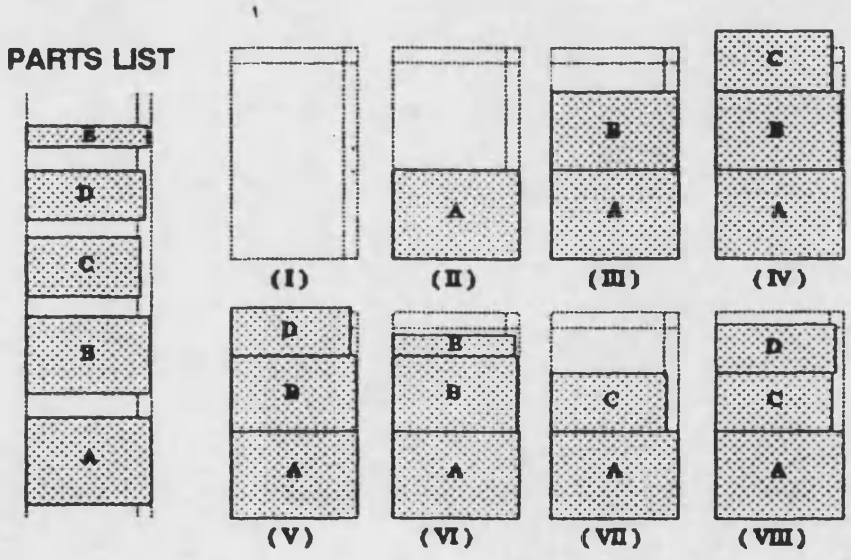


Diagram 2.

All Envelopes with a dimension within the strip width tolerance are removed from the main parts list and placed in a sub group in order of decreasing height, as shown above (working from the bottom up).
(I) Shows the length and width tolerances for the strip.
(II) and (III) Envelopes A and B are placed in the strip and, in each case, it is checked that they do not exceed or meet the strip length tolerance.
(IV),(V) and (VI) Envelopes C, D and E are tried, but none meet the strip length tolerance.
(VII) To allow further envelope combinations to be tried, envelope B is removed and replaced by envelope C. There is no need to check the strip length tolerance as the height ordering of the sub group assures that B is not bigger than C.
(VIII) Envelope D is added and a combination which meets the strip length tolerances is found. If there are other combinations which meet the tolerance they would not include as many large components, however they may provide a more efficient layout. This is accepted to avoid the need for exhaustive trials of all envelope combinations.

Diagram 3 shows a completed 'Strip Forming Dimension Search' layout. The broken lines represent the acceptance tolerances for the length and width of the strips. The system tries to form wide strips first in order to place the larger components as soon as possible. Within each strip the combinations start with the largest Y dimensions envelopes and work towards the envelopes with smaller Y dimensions. This aims to nest the largest components first by assuming that all the X dimensions are approximately the same. It also allows the first strip found meeting the length requirements to be accepted. It is preferential to nest large components early as they are more awkward to use as pattern 'fill in'. The system then checks the remaining envelopes in the sub-group for other satisfactory combinations. If none are found those remaining are returned to the parts list, a new strip width range is set and envelopes with a dimension within this range are searched for.

A Layout created by the 'Strip Forming Dimension Search' Algorithm.

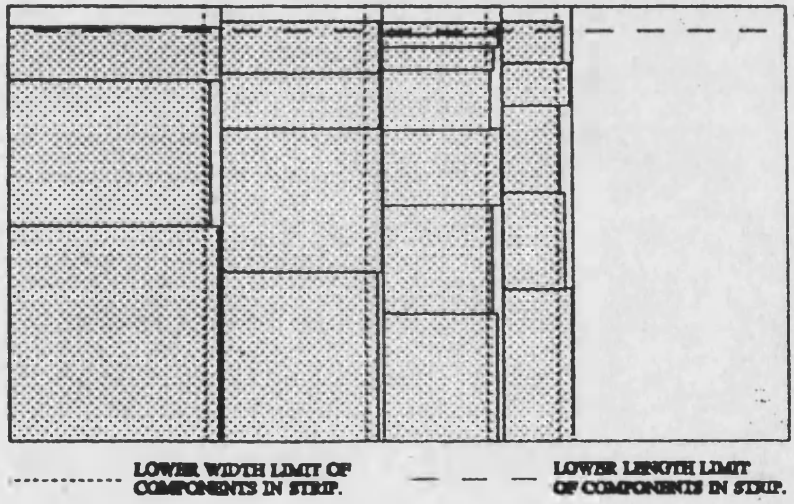


Diagram 3.

Envelopes not utilised to form a strip with the approximate width of their larger dimension will be evaluated a second time for a strip with the approximate width of their smaller dimension. The exceptions to this are if the enclosed component is specified as having a fixed orientation, or if the envelope is a square. If this system is used with tight strip length and width tolerances an efficient layout is produced. However, due to the nature of this heuristic's operation not all the components in the parts list will be nested. If the length and width tolerances are relaxed and this heuristic is used again the remaining components can be nested, however the layout produced would be very inefficient. Alternatively a different heuristic could be employed to nest these remaining component envelopes.

The Strip Squeezing Algorithm.

The 'Strip Forming' algorithm generates a layout which conforms to the guillotine cut constraint i.e. the parts can be cut from the sheet using straight line cuts across the sheet's span. This constraint is often not required with modern cutting methods (laser and rectangular shears) and the layout efficiency can only be improved by its removal. Rather than developing a completely new algorithm to generate an unconstrained layout, an improvement algorithm could be applied to the existing layout. Such an algorithm has been developed and it enhances the 'Strip Forming' algorithm's layout by squeezing pairs of strips together. It should be noted that no additional improvement in this section of the system will be seen when component rotation is permitted as the orientations are inherited from the 'Strip Forming' algorithm. The detailed operation of this algorithm is shown below in diagram 4.

The Stages of the 'Strip Squeezing' Algorithm.

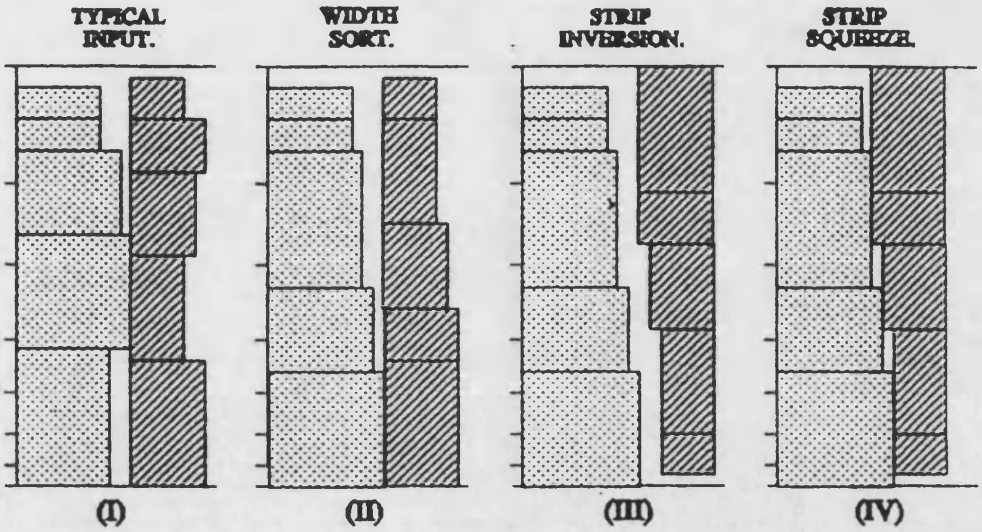


Diagram 4.

(I) Shows two strips conforming to the 'guillotine cut constraint' which have been generated by the 'Strip Forming' algorithm; this is the layout which the 'Strip Squeezing' Algorithm receives. The parts are in decreasing Y dimension order, with the largest Y dimensions at the bases of the strips.
(II) The parts making up each strip have been re-arranged into decreasing X dimension order, again working from the bases of the strips.
(III) The second operation of this algorithm is to invert the strip and align the parts to the right, this corresponds geometrically to a 180° rotation.
(IV) Finally the strips can be squeezed together, reducing the area of sheet which they occupy. The smallest gap between the strips is calculated. This is the maximum translation of the right strip which can be performed without part overlap occurring.

Area Fitting Dimension Search.

This heuristic is far more suited to smaller parts lists containing great variations in component size and shape, as will remain after the 'Strip Forming' algorithm has been used on a large parts list. It operates by identifying the largest component envelope in the parts list which will fit into a space remaining on the sheet. These spaces are maintained in the form of rectangles, and their areas are recorded as well as their dimensions. The envelopes to be nested are ordered by decreasing area.

The system commences its search at the first component envelope of equal or less area than the available space. The dimensions of the component's envelope are then checked to ensure it will fit into the space. As the envelopes are in order of size the first match found will be the best possible. Any area remaining from this process will be used for further nesting, providing suitable components exist in the parts list. If the space is too large i.e. no envelope shares a dimension with it, the largest component envelope in the parts list will be located in the bottom left of the space. From the corner of this envelope either a horizontal or vertical line can be drawn to divide the remaining area into two rectangles. The line which gives the smaller area will be used. This is the primary area to nest. Thus the largest rectangular area possible is kept unused at any one time. A step by step illustration of this algorithm's operation is given in Diagram 5 and a full sheet layout is shown in Diagram 6.

The Operation of the 'Area Fitting Dimension Search' Algorithm.

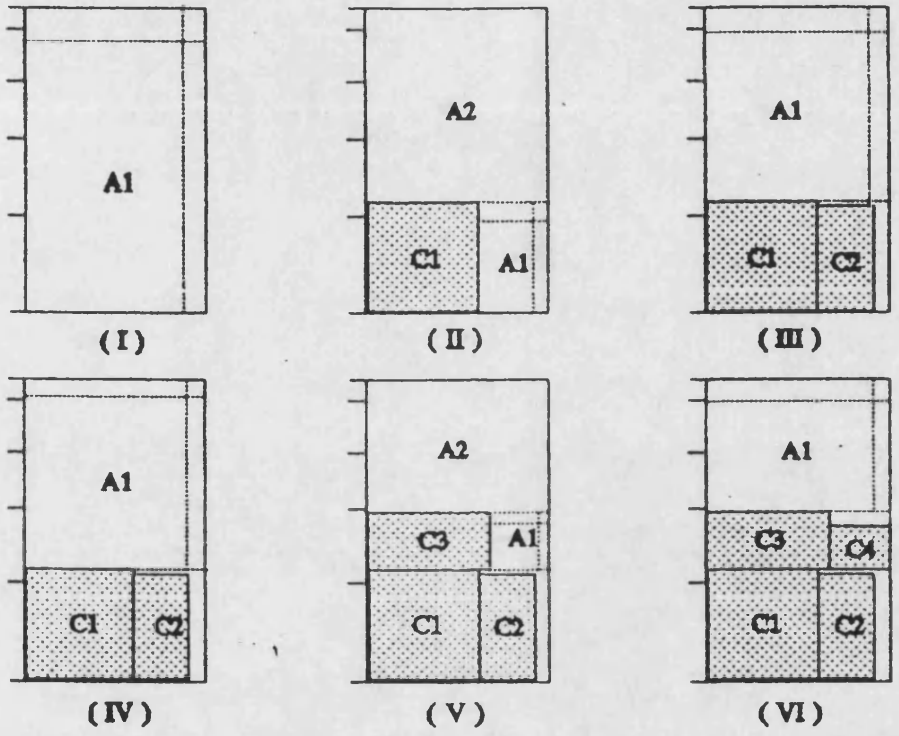


Diagram 5.

(I) Shows the remaining area at the end of a sheet after the Strip Forming heuristic has been used. The dotted lines show the size tolerances, one of which a component must meet to be initially placed in the space.

(II) If no component is large enough to meet the tolerances, as in this case, a contingency rule is used. This is to place the largest envelope from the parts list , C1, in the bottom left corner of the space. The remaining area is now divided into A1 (the primary area to nest) and A2 (the remaining area). The area could have been divided using a vertical line from the top right corner of C1. However, it is considered better to maximise the size of the 'largest remaining area' at all times.

(III) Envelope C2 fits A1.

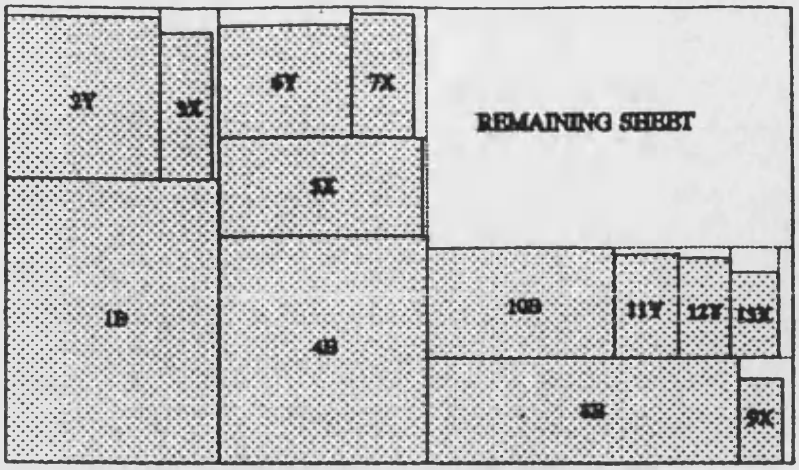
(IV) No envelope fits the new A1.

(V) Largest remaining envelope C3 is placed creating new spaces A1 and A2.

(VI) Envelope C4 fits A1.

Note: A1 and A2 represent the first two positions in a 'stack' of areas to be filled, which is why their values change. In this example the stack is never loaded with more than two areas.

A Layout created by the 'Area Fitting Dimension Search' Algorithm.



NUMBERS INDICATE ORDER OF COMPONENT NESTING.
LETTERS INDICATE HOW THE COMPONENT WAS PLACED.
B = 'BOTTOM LEFT' CONTINGENCY. X = X DIM. FIT. Y = Y DIM. FIT.

Diagram 6.

Testing of the system.

Isolated testing of the new system would be meaningless as the general materials efficiencies reported in industry are not for purely rectangular parts and they vary greatly. Direct comparison with human planners or commercial systems would require the rectangular enveloping 'front end' to be developed. Therefore, at this stage the most meaningful test is to compare the new system against the existing 'intervention free' algorithms for rectangular parts. The two algorithms to be used as benchmarks are the 'Next-Fit Decreasing-Height' and 'First-Fit Decreasing-Height' algorithms (Coffman[5]) which are described above. As both of these algorithms are suitable for improvement with the 'Row Squeezing' algorithm these combinations are to be tested. Both the 'Strip Forming Dimension Search' and 'Area Fitting Dimension Search' algorithms developed can be used individually and are also to be tested.

In order to evaluate the efficiency of a layout it is necessary to define what will be considered as waste. With completely nested sheets there is rarely any doubt that all the unused sheet is waste, although occasionally a particularly large area may remain which would be worth retaining for future use. On part filled sheets, defining what of the remaining area can be considered useful stock material is more difficult. One method is to take the area which the algorithm could have continued to use, had the parts list been larger. However, for simple stock management, only large rectangular sections would be retained for future use, so the above method of evaluation may not always reflect practical material savings. Insufficient tests have been carried out to give the average performance of each algorithm accurately. The maximum and minimum percentage nesting efficiencies found so far for each algorithm are given in Table 1. In addition to performing better, the new algorithms are also more consistent.

The absolute performance of a nesting algorithm is hard to define as some waste, such as the 'bridge gap', cannot be avoided. One approach is to create a parts list with a known benchmark layout which cannot be improved. The layout structure must not allow any test algorithm to duplicate it. The layout used (Diagram 7) has 25 parts, separated by a 0.02m bridge gap, occupying 60% of a 2.5m X 1.5m sheet. Layouts generated by the benchmark algorithms (Diagrams 8 & 9), the 'Strip Forming' and 'Strip Squeezing' algorithms (Diagram 10), the 'Area Fitting' algorithm (Diagram 11) and the total system (Diagram 12) are shown.

Upper and lower performance levels
for the 8 nesting algorithms tested.

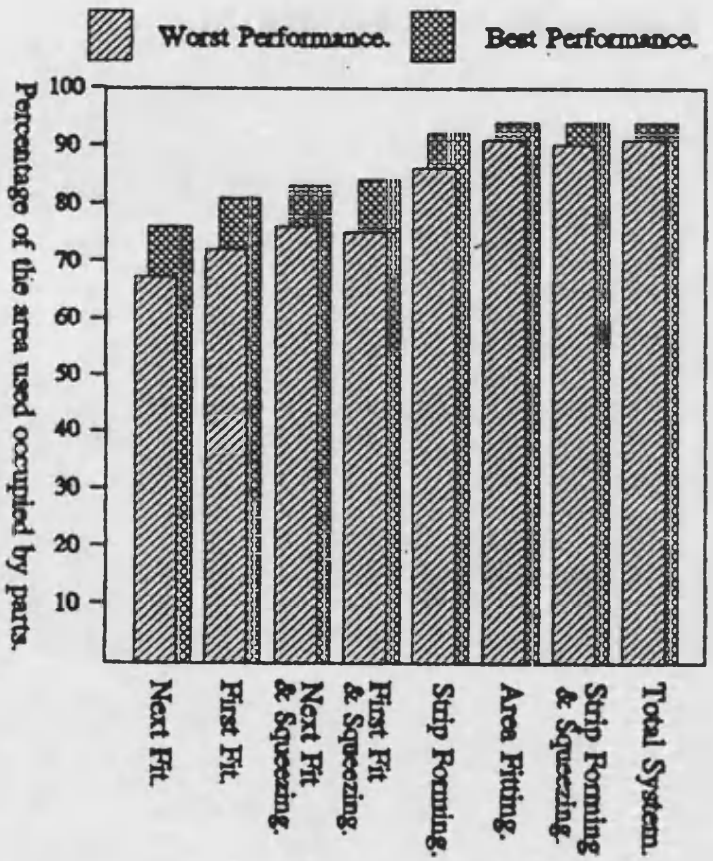


Table 1.

The Perfect Layout for the Part List.

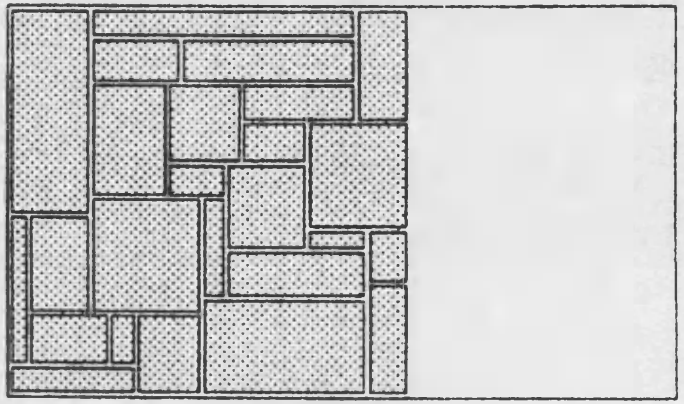


Diagram 7.

The Next-Fit Decreasing-Height Layout.

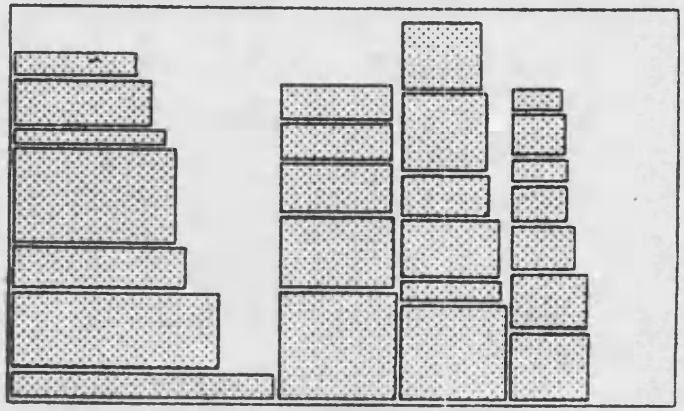


Diagram 8.

The First-Fit Decreasing-Height Layout.

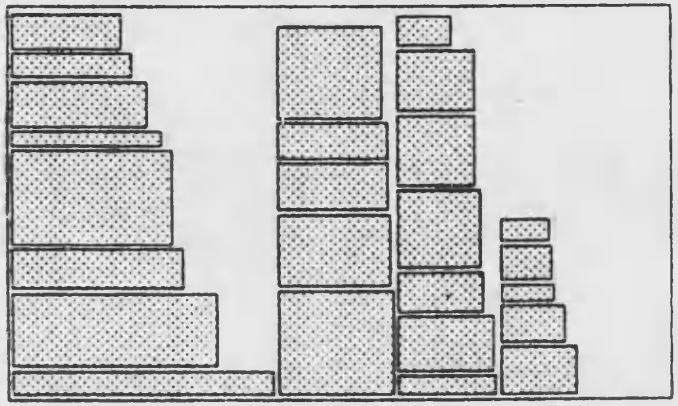


Diagram 9.

The Strip Forming And Squeezing Layout.

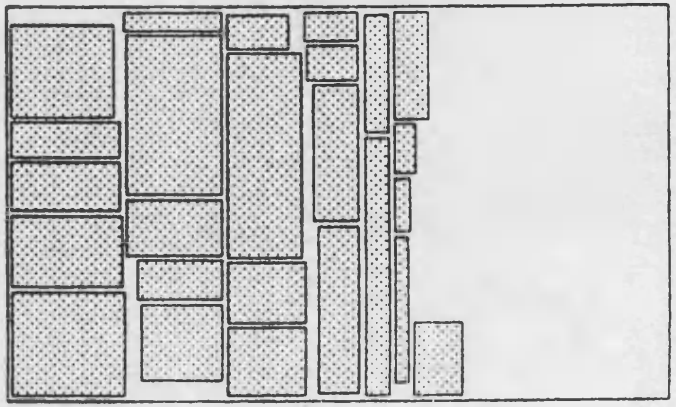


Diagram 10.

The Area Fitting Layout.

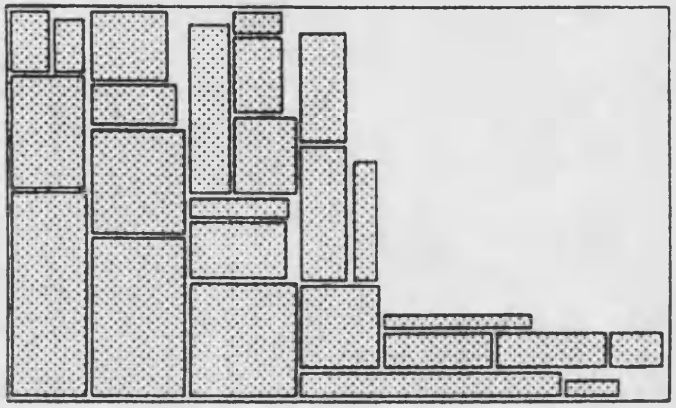


Diagram 11.

The Total System Layout.

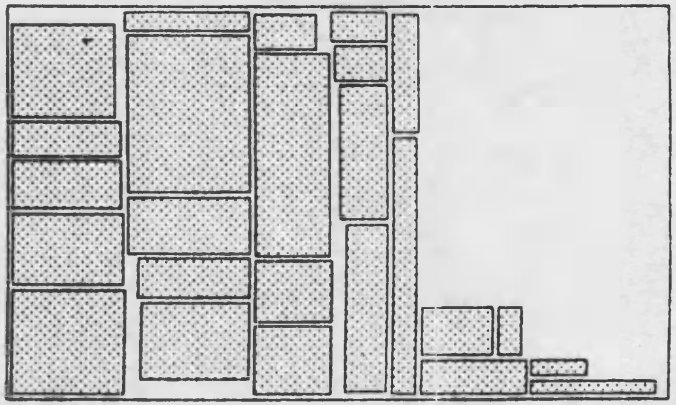


Diagram 12.

Conclusion.

This paper has examined the nesting of sheet metal parts onto prescribed sizes of stock sheet. A review of existing nesting systems has been carried out and their limitations have been identified. A novel nesting system is described, based on enclosing parts within rectangular envelopes, which uses three novel algorithms sequentially. Unlike many existing algorithms, the new system uses rules to choose the best component to place at each stage. Thus the sequence of component placement is evolved during the nesting process. The initial tests give cause for optimism. The new algorithms have not been out performed by the existing algorithms for any test parts lists tried. In addition, for many of the test parts lists the new system's results have been radically better than existing systems.

Further work.

A comprehensive set of structured tests are required to accurately evaluate the performance of the new algorithms against the established algorithms. The work of Israni and Sanders[6] has highlighted the effects on layout efficiency of the part aspect ratio (length / width) and the part size : sheet size ratio. In addition, the size of the parts list, the sheet aspect ratio and whether the parts may be rotated may also affect the nesting performance. The effects of these characteristics will be investigated to establish the robustness of each algorithm to them.

Both the 'Strip Forming Dimension Search' and 'Area Fitting Dimension Search' algorithms require part size acceptance limits to be set. These represent the greatest waste around a part which the algorithm should accept and still place components. The optimum levels of these acceptance limits cannot be calculated and therefore must be established by experimentation. These optimum levels may also vary according to the characteristics of the parts to be nested and the stock sheet used.

Acknowledgements.

The research work carried out towards this paper has been funded by SERC.

References.

1. Gilmore P.C. and Gomory R.E. A Linear Programming approach to the Cutting Stock Problem. Opns. Res., 9, 849-859, 1961.
2. Gilmore P.C. and Gomory R.E. Multi-stage Cutting Stock Problems in two or more dimensions. Opns. Res., 13, 94-128, 1965.
3. Halms J.M. and Freeman H. A multistage solution of the template-layout problem. IEEE Trans. Syst. Sci. and Cybernetics, Vol. SSC-6, No. 2, April 1970.
4. Baker B.S., Coffman E.G.Jr. and Rivest R.L. Orthogonal packing in two dimensions. SIAM J. COMPUT. Vol. 9, No. 4, Nov 1980.
5. Coffman E.G.Jr., Garey M.R., Johnson D.S. and Tarjan R.E. Performance bounds for level-oriented two-dimensional packing algorithms. SIAM J. COMPUT. Vol. 9, No. 4, Nov 1980.
6. Israni S.S. and Sanders J.L. Performance testing of rectangular parts-nesting heuristics. Int. J. Prod. Res. Vol. 23, No. 3, pp 437-456, 1985.
7. Adamowicz M. and Albano A. A solution of the rectangular cutting-stock problem. IEEE Trans. Syst. Man. And Cybernetics, Vol. SMC-6, No. 4, April 1976.
8. Qu W. and Sanders J.L. A nesting algorithm for irregular parts and factors affecting trim losses. Int. J. Prod. Res. Vol. 25, No. 3, pp 381-397, 1987.
9. Adamowicz M. and Albano A. Nesting two-dimensional shapes in rectangular modules. Computer Aided Design. Vol. 8, No. 1, Jan 1976.
10. Dagli C.H. and M. Tatoglu M.Y. An approach to two-dimensional cutting stock problems. Vol. 25, No. 2, pp 175-190, 1987.
11. Tiow O.K., Wah P.W. and Guang W.C. Optimisation of materials processing using AutoCAD. J. Mats. Proc. Tech, 29, pp 293-2999, 1992.
12. Scott A.J. and Mileham A.R. Automated Nesting of Sheet Metal Components onto Stock Sheets. Advances in Manf. Tech. VII. pp 242-246. University of Bath, Sept. 1993.